63-3-2

AMRL-TDR-63-17

402 383

DYNAMIC TASK SCHEDULING IN FLIGHT SIMULATORS

TECHNICAL DOCUMENTARY REPORT NO. AMRL-TDR-63-17

February 1963

Behavioral Sciences Laboratory
6570th Aerospace Medical Research Laboratories
Aerospace Medical Division
Air Force Systems Command
Wright-Patterson Air Force Base, Ohio

Contract Monitor: Arthur B. Doty, Jr.
Project No. 6114, Task No. 611409

[Prepared under Contract No. AF 33(616)-8062
by
Jerome M. Kurtzberg
Burroughs Corporation, Burroughs Laboratories, Paoli, Penna.]

NOTICES

25-594, 800, 4-22-63

Aerospace Medical Division,
6570th Aerospace Medical Research
Laboratories, Wright-Patterson AFB, Ohio
Rpt. No. AMRL-TDR-63-17. DYNAMIC TASK SCHEDULING IN FLIGHT SIMULATORS. Final report, Feb 63, iv + 45 pp, incl. illus., table, 13 refs. Unclassified report

This report deals with the possible mechanization of dynamic task scheduling in flight simulators, i. e., developing a Task-Sequencer. Attention is focused on the possible application of some of the heuristic programming techniques and an evaluation of their worth for that specific purpose is made. Two main applications for a Task-Sequencer are defined. The first involves the traditional training of students (flight crews) for flight vehicle operation, termed (over)

the operation-teaching mode. The second is for the development of tactical skill, i. e., crew decision-making capabilities, termed the tactic-teaching mode. Algorithms for task sequencing in real time are formulated for both of these classes of applications. The only possible benefits in employing a heuristic programming scheme appear to exist when it is used for an ancillary role in the tactic teaching mode. This includes development of specific task flow diagrams and associated scoring charts. Finally, recommendations are made for further work.

UNCLASSIFIED

1. Flight Simulators
2. Training and Training Aids
3. Procedural Tasks
4. Simulation (Aerodynamics)

I. AFSC Project 6114, Task 611409
II. Behavioral Sciences Laboratory
III. Contract AF 33(616)-8062
IV. Burroughs Corporation, Burroughs Laboratories, Paoli, Pennsylvania

UNCLASSIFIED

UNCLASSIFIED

V. J. M. Kurtzberg
VI. In ASTIA collection
VII. Aval fr OTS: $1.50

UNCLASSIFIED

Aerospace Medical Division,
6570th Aerospace Medical Research
Laboratories, Wright-Patterson AFB, Ohio
Rpt. No. AMRL-TDR-63-17. DYNAMIC TASK SCHEDULING IN FLIGHT SIMULATORS. Final report, Feb 63, iv + 45 pp, incl. illus., table, 13 refs. Unclassified report

This report deals with the possible mechanization of dynamic task scheduling in flight simulators, i. e., developing a Task-Sequencer. Attention is focused on the possible application of some of the heuristic programming techniques and an evaluation of their worth for that specific purpose is made. Two main applications for a Task-Sequencer are defined. The first involves the traditional training of students (flight crews) for flight vehicle operation, termed (over)

the operation-teaching mode. The second is for the development of tactical skill, i. e., crew decision-making capabilities, termed the tactic-teaching mode. Algorithms for task sequencing in real time are formulated for both of these classes of applications. The only possible benefits in employing a heuristic programming scheme appear to exist when it is used for an ancillary role in the tactic teaching mode. This includes development of specific task flow diagrams and associated scoring charts. Finally, recommendations are made for further work.

UNCLASSIFIED

1. Flight Simulators
2. Training and Training Aids
3. Procedural Tasks
4. Simulation (Aerodynamics)

I. AFSC Project 6114, Task 611409
II. Behavioral Sciences Laboratory
III. Contract AF 33(616)-8062
IV. Burroughs Corporation, Burroughs Laboratories, Paoli, Pennsylvania

UNCLASSIFIED

UNCLASSIFIED

V. J. M. Kurtzberg
VI. In ASTIA collection
VII. Aval fr OTS: $1.50

UNCLASSIFIED

FOREWORD

This report was prepared by the Burroughs Corporation for the 6570th Aerospace Medical Research Laboratories. The original study work upon which the report is based was accomplished by the Burroughs Laboratories of Paoli, Pennsylvania, under Air Force Contract No. AF 33(616)-8062, Project No. 6114, "Simulation Techniques for Aerospace Crew Training," Task No. 611409, "System Synthesis." The author, Jerome M. Kurtzberg, was the project engineer in charge of the study and developed the algorithms. Dr. Joel Morris aided in the collection of data for the procedural model and developed the specific task flow diagrams for the model shown in this report. Arthur B. Doty, Jr., Simulation Techniques Section, Training Research Branch, Behavioral Sciences Laboratory, served as contract monitor.

Acknowledgment is made of the assistance provided by the personnel of the Behavioral Sciences Laboratory, 6570th Aerospace Medical Research Laboratories, and the personnel attached to the KC-135 Simulator (SAC), Wright-Patterson Air Force Base, Ohio.

We wish to thank the flight training instructors, in particular Captain Geoffrey Brown and Lieutenant Richard Crumbliss, for their consideration and aid in the development of a realistic procedural model for our study, and Mr. Morris Siskel of Bell Aerosystems Company for stimulating discussions with the author at Castle Air Force Base, Merced, California.

ABSTRACT

This report deals with the possible mechanization of dynamic task scheduling in flight simulators, i.e., developing a Task-Sequencer. Attention is focused on the possible application of some of the heuristic programming techniques and an evaluation of their worth for that specific purpose is made.

Two main applications for a Task-Sequencer are defined. The first involves the traditional training of students (flight crews) for flight vehicle operation, termed the operation-teaching mode. The second is for the development of tactical skill, i.e., crew decision-making capabilities, termed the tactic-teaching mode.

Algorithms for task sequencing in real time are formulated for both of these classes of applications. The only possible benefits in employing a heuristic programming scheme appear to exist when it is used for an ancillary role in the tactic-teaching mode. A procedural training model is developed in detail for the operation-teaching mode. This includes development of specific task flow diagrams and associated scoring charts. Finally, recommendations are made for further work.

PUBLICATION REVIEW

This technical documentary report has been reviewed and is approved.

Walter F. Grether

WALTER F. GRETHER
Technical Director
Behavioral Sciences Laboratory

TABLE OF CONTENTS

SECTION I

GENERAL DISCUSSION

INTRODUCTION

The purpose of this study was to investigate the possibility of efficiently mechanizing some of the functions which are presently performed by the instructor during training sessions in flight simulators. Of particular interest was an investigation to determine the feasibility and desirability of using heuristic programming techniques for this purpose. This relatively new technique has become quite popular in simulating some of the decision processes of humans. However, if heuristic programming was not found applicable, algorithms were to be derived by the use of other techniques.

Specifically, the instructor's function that was selected for study was the sequencing of tasks for presentation to the student pilots or crews in flight simulators - that is, developing a mechanized Task-Sequencer that relieves the instructor from sequencing tasks, thereby freeing him for more effective instruction.

In present flight simulators, instructors train students by selecting tasks in the form of emergency conditions, having them inserted into the simulator from a large panel of switches and potentiometers, and then monitoring, recording and correcting, if necessary, the student's performances. The non-emergency tasks normally associated with operating a flight vehicle are expected to be performed at the appropriate time under the initiation of the student. The defects of this method of instruction are: (1) the instructor is forced to concentrate upon the mechanics of selection and presentation of the tasks and in scoring of the student's performance, (2) large areas in flight simulators are required to house the instructor's equipment, and (3) frequently, there is insufficient and inaccurate data for feedback to the student and for use in later training (both for the particular student and for other students).

Future simulators will amplify the above problems due to the increased complexity and performance capabilities of the vehicles that will be simulated. For these simulators it is important to have a mechanized system that will automatically sequence tasks. This involves selecting and presenting tasks, comparing performance with established criteria and recording the results, and deciding to select new or rerun old tasks, depending on the student's past performance. With such a system the instructor will be free to concentrate on correcting the errors of the student and have additional time to give them more personal attention.

In order to arrive at proper conclusions and to formulate the task sequencing algorithms, some flight simulator sites were visited, and a study and an analysis were made of existing training techniques by consultation with trained flight instructors and by detailed examination of various flight manuals. These empirical investigations provided the foundation and rationale for much of the work presented in this report.

Algorithms for real-time use in various modes of the Task-Sequencer were developed which satisfy the condition that the selection of tasks be based upon the specific performance of the student on previous tasks assigned to him, taking into account his specific level of training and experience. These algorithms are discussed in detail in section III. Essentially there are two different modes of operation for the Task-Sequencer: one is for the teaching of procedural tasks for the operation of flight vehicles; the other is for training crews in tactical operations. These modes are termed respectively the operation-teaching mode and the tactic-teaching mode of the Task-Sequencer and are discussed later in the text.

Because the operation-teaching mode is the primary concern of present-day flight simulators, the largest part of the effort in this study was concentrated on this mode. Although the tactic-teaching mode, as defined within this report, is not currently used for training crews, it appears highly desirable and useful; however, due to the time limitations and the scope of the project, it was not examined with the same detailed attention.

Since this report is intended for a wide audience, an attempt has been made in the presentation of the conclusions and the explanation of the task sequencing methods to simplify whenever possible all explanation of concepts. Therefore, background material on flight training techniques is included in this first section, as well as a discussion on the application of the Task-Sequencer for various training models, and an evaluation of heuristic programming for the Task-Sequencer.

Furthermore, many terms may be used which possibly have specific military connotations differing from the general usage of the terms. When this has occurred an attempt has been made to specifically spell out the meaning intended, either by explicitly defining the term or by giving examples later in the text. For example, tactic teaching should be taken as synonymous with teaching of crew decision making in a tactical situation.

In addition, for clarification within the report, the following explanation of heuristics is presented. Heuristic programming, sometimes contracted to heuristics, is a technique for delimiting a sample space of a large universe of candidate solutions to a given problem so that an impractical exhaustive examination of the sample points is not necessary.

For example, there may be 10^{10} points and only sufficient time to examine 12. Thus it is important to judiciously choose the most promising candidate points.

The rules for selecting the trial sample points to be considered for the solution are the heuristics. The choice of the specific heuristics is strongly dependent upon the type of problems to be solved -- different problems, different heuristics. Frequently, the inspiration or rationale for these heuristics is derived from observation of the approach humans employ in solving problems of a similar nature. Very often, the approach consists of setting up sub-goals or sub-tasks which one solves in the expectation that this will lead to solution of the main goal; the heuristics suggests the sub-tasks.

A solution, or optimal solution, is not guaranteed; the heuristics may exclude from consideration the desired solution. Essentially, heuristic programming is an approximation method useful where no known solution method exists.

An evaluation of heuristics, as a result of this study, for the sequencing of tasks is presented later in this section.

In section II a procedural model is defined in detail complete with flow diagrams of 14 specific tasks for the operation-teaching mode of the Task-Sequencer. The meaning of a procedural model is defined in the beginning of that section.

In section III algorithms are presented for automatic task sequencing. Also included is a discussion of the structure of a heuristic program.

Finally, section IV contains the conclusions along with recommendations for further work.

APPLICATIONS OF TASK-SEQUENCER

There are two main applications for flight simulators, and thus for a Task-Sequencer. The first, the operation-teaching mode, involves the traditional teaching of students -- both novice and combat ready pilots -- to operate their vehicle and to detect and correct any malfunctions of their craft. In the case of the combat ready pilot, the teaching may be only of the form of a periodic test or check on their skill with reinforcement of their already learned techniques. With a novice, these techniques or skills will have to be taught. In both cases, however, what is being taught is the operation of a vehicle; that is, the acquisition of vehicle operation skill by learning procedural operations. The algorithm described in section III, which is for the operation-teaching mode of the Task-Sequencer, is designed for this application.

The second application, the tactic-teaching mode, has a more interesting although more difficult purpose; that is, teaching students (or crews) tactical operations -- decision making in a tactical situation. This mode, as examined from a game theoretic viewpoint may be looked upon as a two-person, zero-sum game with nature playing the student's opponent. It permits the student, when presented a complex tactical situation, to try various strategies and observe the effects; thus, it provides a painless way of acquiring "practical" experience.

The tendency for flight vehicles to become more and more automatic with the pilot possessing over-ride capabilities infers that the operation of these vehicles will, in many respects, be similar to the management of a corporation in which action is only taken on the exceptional case. Such a procedure is sometimes termed management-by-exception. Likewise, the student deciding which particular subsystem to over-ride or monitor, with the costs associated with each decision and the benefits to be derived from the correct choice, constitutes a game (in the mathematical sense) similar in intent to the various management system computer simulations. It seems present flight simulators do not adequately fulfill this function as they are mostly concerned with developing skill in procedural operations.

Also timely and important, as it bears upon the present problem of training Stategic Air Command (SAC) crews for optimal performance of their assignments, is the penetration tactics of SAC. The ECM (electronic counter-measures) man can be considered to be directly opposed by his enemy counter-part on the ground, the ECCM. (The ECM has to protect his vehicle against surface-to-air and air-to-air threats; for example, he must decide when to and when not to jam radio frequencies in order to avoid detection.)

Certainly this active and direct opposition with the enemy also can be cast into the guise of a two-person zero-sum game, for which the algorithm for the tactic-teaching mode of the Task-Sequencer, given in section III, is applicable.

EVALUATION OF HEURISTICS FOR THE TASK-SEQUENCER

Of major importance in the investigation of the application of heuristics for the Task-Sequencer is the amount of influence that a response of the student to a given task has on the next task chosen by the instructor.

A heuristic programming technique would be most useful in a situation where there is a large number of possible tasks that could be presented to the student. To choose an appropriate subset of these tasks would call for certain tasks to have common characteristics, and rules (heuristics) to distinguish them.

However, in existing simulators the tasks given to the student by the instructor are taken from a check-list (the lesson plan), usually in sequential order. There is a given number of tasks, all of which must be learned; the student must be, and is, tested on all of them. There is seldom an attempt to present only a sampling of some representative tasks, and conditional upon the response of the student, to omit most of the other tasks of a similar nature. About all that is done is to modify the presentation of stimuli or malfunction symptoms to the student. For example, these symptoms might range from a few dials moving from their normal position to actual smoke being sent into the simulated cockpit. A good student would receive the more subtle signs of danger.

Essentially, the simulator operator or the instructor inserts malfunctions according to a fixed lesson plan, and the student must first apply diagnostic techniques, then corrective procedures. These corrective procedures are given to the student in advance as a series of predetermined steps. He has these check-list procedures in a booklet on his lap and is expected to go thru them in a methodical manner after the diagnosis of trouble. The emergency tasks have been selected by considering common precautionary measures and by performing extensive investigations and analyses of flight malfunctions and actual aircraft crashes. Since the Air Force has had considerable experience with the aircraft being simulated, the list of emergency tasks is surprisingly complete.

By using a fixed lesson plan the instructor cannot allow for conditional interplay nor can he consider the student's previous response when selecting the next task; furthermore, the order of these tasks, as presently set up, reflects little or no progressive difficulty. A flight simulator and a training technique which exemplifies these characteristics will be discussed later.

It should be stressed, however, that adaptive devices were not available to past designers of flight simulators or to present flight instructors. The instructors are constrained to teaching patterns that are not dependent upon adaptive devices. Hence, the structure of the present flight simulators and instruction techniques is strongly biased toward a straightforward task selection independent of the student's previous performance.

The scheme presented for the operation-teaching mode of the Task-Sequencer has provisions for including heuristics to aid in the selection of tasks, especially if the range of tasks is broadened to include an optional class of tasks as well as the mandatory tasks. However, with the present teaching scheme, namely the fixed list of mandatory tasks to be both learned and tested, there does not appear to be sufficient benefit in a heuristic programming scheme in the sense of Newell, Shaw and Simon (ref. 1) for sequencing tasks.

The scheme proposed for the operation-teaching mode is adaptive in the sense that the scheduling of tasks proceeds by means of a form of statistical averaging and extrapolation of trends for each specific student; it is not necessary to have a predetermined fixed order of presenting tasks. Since the technique also provides for the inclusion of heuristics, if deemed necessary at a latter point, it appears worthy of actual implementation as a practical technique.

The use for heuristic programming in flight simulators would possibly be an auxiliary role in the tactic-teaching mode of the Task-Sequencer, in the dynamic case. This problem, as already discussed, is akin to the computer simulated management systems and is not concerned with teaching detailed vehicle procedural operations.

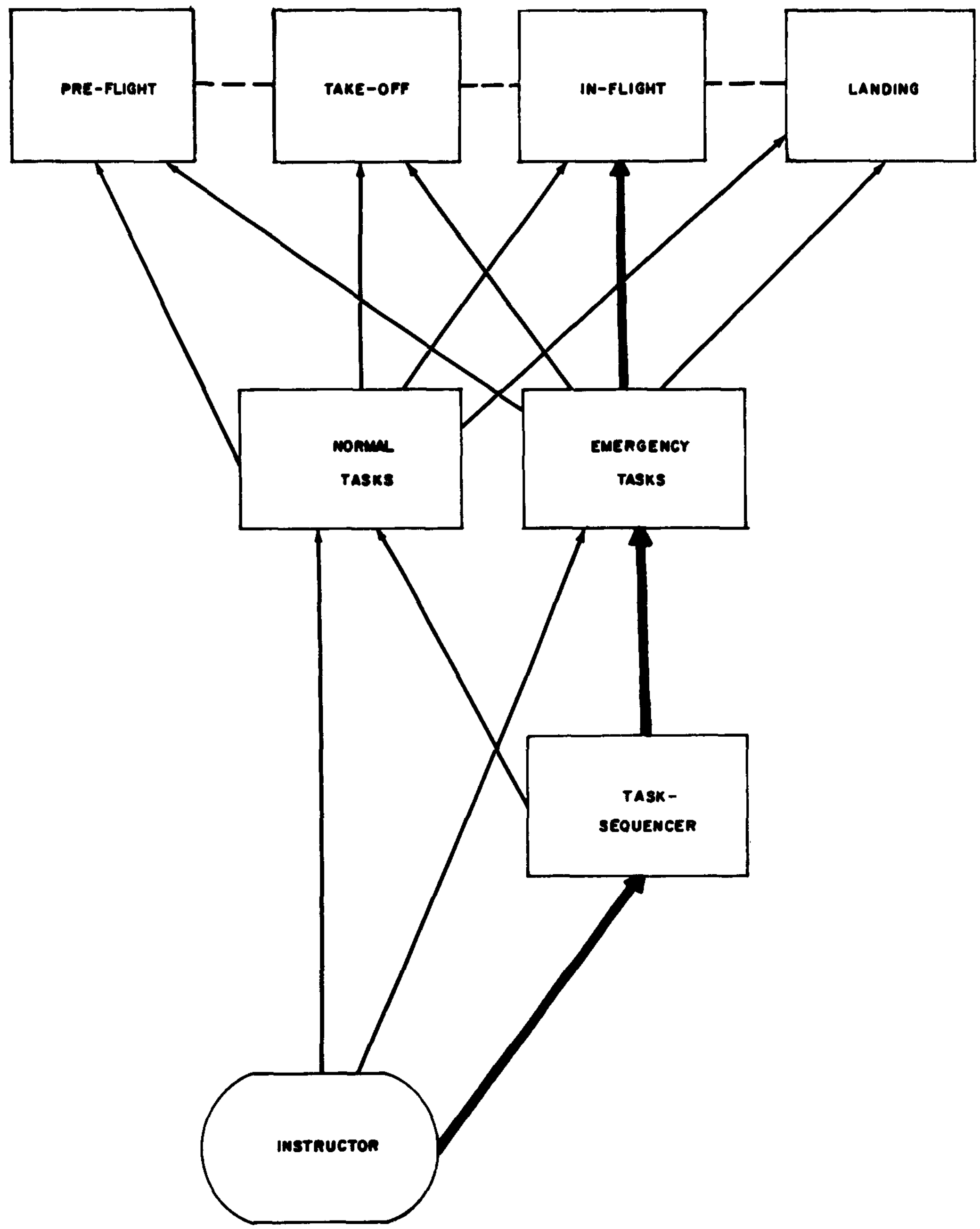
The algorithm given in section III for sequencing tasks in the tactic-teaching mode appears quite adequate. Although it does not resort to a heuristic scheme in the formal sense, there would be an application for such a scheme within the proposed algorithm. However, the heuristic scheme would not be needed for sequencing tasks; its only application would be in the modification of the entries in the pay-off matrix. The pay-off matrix is the representation of the pay-off, or value, of various choices of tactics or strategies on the part of the student (crew) and the opponent. For a realistic representation, in the general case these entries may be nonstationary distributions which should be modified after each task cycle. That point is discussed in section III. A heuristic scheme operating upon the last specific position choice of the Task-Sequencer, and the student, appears to provide a capable means of handling the pay-off matrix modification during real-time play.

DOMAIN OF ACTIVITY FOR TASK-SEQUENCER IN OPERATION-TEACHING MODE

There are certain fairly well segregated functions, and associated procedural operations, for all flight vehicles which pilots must learn. For convenience these can be logically partitioned into four main areas: (1) pre-flight, (2) take-off, (3) in-flight, and (4) landings. Each of these areas have relatively diverse sets of tasks for the student. Furthermore, there are two classes of these tasks, "normal" and "emergency." The student is expected to perform "normal" tasks under his own initiative; "emergency" tasks are the ones the instructor presents to the student. In present flight simulators, the instructor serves as a monitor for both classes of tasks and informs the student of the correctness and appropriateness of his responses. He is also quite active in presenting tasks, simulating the rest of the crew, and in supplying correct procedures.

Figure 1 illustrates the relationship of the Task-Sequencer to a normal task selection structure. It should be noted that the Task-Sequencer is under the control of the instructor, and is intended to relieve him of sequencing tasks -- not to replace him. The domain of activity of the Task-Sequencer in the proposed model is in choosing in-flight, emergency tasks, to be presented to the student. That choice conforms to the expressed desires of flight instructors as the most useful area for initial mechanization. A list of the chosen tasks is given on page 7. Reference to section II will supply an understanding of these tasks as defined in this report for the operation-teaching mode.

It is not difficult to generalize the control of the Task-Sequencer to include the other areas, such as pre-flight, take-off, and landings, as well as the "normal" class of tasks. A complete mechanization of all the procedural task areas would merely require the development of additional task flow diagrams with associated task evaluation values. It is immaterial if the total number of procedural tasks for all the operational areas is held in the memory of the computer simultaneously, or if the procedural tasks for each area are individually contained with consequent re-loading of memory for each area. That decision for the manner of mechanization must be made on the basis of the amount of internal memory available.



THE PATH SHOWN BY THE HEAVY LINES GIVES THE AREA OF CONCENTRATION FOR THE OPERATION-TEACHING MODE DEVELOPMENT AND FEASIBILITY STUDY.

Figure 1 - Task Selection Flow Diagram

SECTION II

PROCEDURAL MODEL FOR OPERATION-TEACHING MODE

In order to simulate an operational system, a detailed analysis must be made to define the system and to acquire familiarization with all aspects of its operation. The information obtained from this analysis is used to generate a model of the system from which a simulation technique can be formulated.

The model developed in this section is for the operation-teaching mode and is delimited to procedures which make up a given task, hence the title procedural model. These procedures are directly related to in-flight emergency tasks (refer to figure 1). The procedural model is defined complete with quantitative values for scoring students.

The development of the procedural model was important not only to furnish a basis for the operation-teaching mode of the Task-Sequencer, but also served to give insight in the present techniques of pilot training and to supply grounds for reasonable extrapolations for the training methods to be used with future flight simulators.

In addition, a realistic procedural model enables one to estimate better the workings and value of the task-sequencing algorithm for the operation-teaching mode. One of the main values, however, for developing a realistic model was to study the possible application of heuristics for the Task-Sequencer. The only way this can be done is by the detailed study of the specific tasks.

MODEL DEFINITION

This particular procedural model developed in this section is intended for a student, in this case, a pilot and copilot.

Fourteen, in-flight, emergency tasks have been flow-diagramed along with measures of performance and ranking of various alternate responses open to the student. These tasks represent a good sampling of realistic problems which a pilot and copilot might encounter during flight.

The Task-Sequencer presents these tasks to the student by appropriate symptoms, e.g., via the simulated vehicle's instrumentation, records the actions of the student, and selects the next task to be presented. Selection of the next task is a function of the response of the student.

In response to any given task, there are various steps or operations comprising the correct performance; however, it is also possible for a student to perform different combinations of the steps and operations, or do most but not all, and still obtain a near-optimum score. It is assumed that the instructor will correct the student for a nonoptimal performance.

For a near-optimum performance on a given task, the student does not deserve to receive a score equivalent to one who has failed miserably. Therefore, a scoring system is set up to evaluate the student's performance by allowing certain scores for performing various combinations of the steps and operations. This the algorithm does automatically. In addition, a subroutine could be created to allow certain time limits for performance in response to any given task. This can be adjusted for the student's relative skill; i.e., more time for a novice as compared to the time allowed a combat-ready student. These matters are discussed in section III which covers the algorithms for the operation-teaching mode of the Task-Sequencer.

For each of the tasks given in this section there is a detailed flow diagram and an associated scoring chart listing the corresponding scores (in percentages) for the correct performance of various combinations of operations. These are presented in figures 3 through 16.

The detailed operations were obtained by studying various flight manuals and consulting with highly trained flight instructors. The flight instructors provided the data upon which the scores for the various combinations of operations for each task are based.

All of these tasks require the pilot and copilot to be fully aware of the operation of the flight controls, the electrical system, and the power system. Since the pilot and copilot are to act as a coordinated team and are conceived as a unit, they have been scored as one student. All of the tasks presented to the student can be perceived from the student's normal operating position with the exception of the circuit-breaker panel.

In tables I and II the various tasks comprising the procedural model are ranked with respect to "importance" and "difficulty" respectively. This was done by the simple expedient of questioning a number of trained and qualified flight instructors. Six flight instructors were consulted to avoid any undue bias due to individual personal preferences.

Briefly, task importance is a measure of the relative value of knowing the emergency procedure. Task difficulty is a measure of how hard or complex the task is, or how long it takes (on the average) to master it. The various percentages shown are quantitative measures of the rank order of the tasks and range between 1 and 100 percent.

These measures of task importance and difficulty are used in the algorithm for the operation-teaching mode of the Task-Sequencer as elements for a weighting factor for the task scores achieved by the students. (See section III).

Interestingly enough, there was little positive correlation among the instructors concerning the measures of task importance and task difficulty. This would mean that while the students must learn the corrective action for a certain set of required tasks, each instructor would emphasize different tasks; that is, an instructor would spend more time on certain tasks than other instructors and thus concentrate on different learning goals.

The detailed flow diagrams with associated task scoring charts are given on pages 11 thru 24 and the pertinent abbreviations for these diagrams are found on page 45. Note some items which appear in these flow diagrams are in a sequentially vertical form. These are termed "killer items" and require instantaneous action on the part of the student. He must have them memorized and practiced to the point of automatic reaction. He does not have enough time to look up the correct procedure in a manual while in flight. For many other tasks he has sufficient time to do this.

The following is a list of the 14 tasks given in the flow diagrams with the figure numbers and an enumeration of the number of variations or combinations of operations for which a certain score is given.

Figure No.	Task	No. of Variations
3	Engine Inlet Icing	5
4	Instrument Failure Due to Pitot Icing	6
5	Surface Icing	3
6	Smoke from Air Conditioner	8
7	Smoke from Communications Panel	6
8	Smoke & Fumes Elimination	6
9	Fuselage Fire	11
10	Engine Fire While Crusing	10
11	Fuel Dumping	7
12	Emergency Descent	9
13	Bailout	4
14	Engine Relight	5
15	Engine Air Start	13
16	Runaway Stabilizer Trim	18
	Total Variations	111

A graphical representation is shown in figure 2 of a typical ordering or sequencing of the above tasks that could occur in a training session with the Task-Sequencer in the operation-teaching mode.

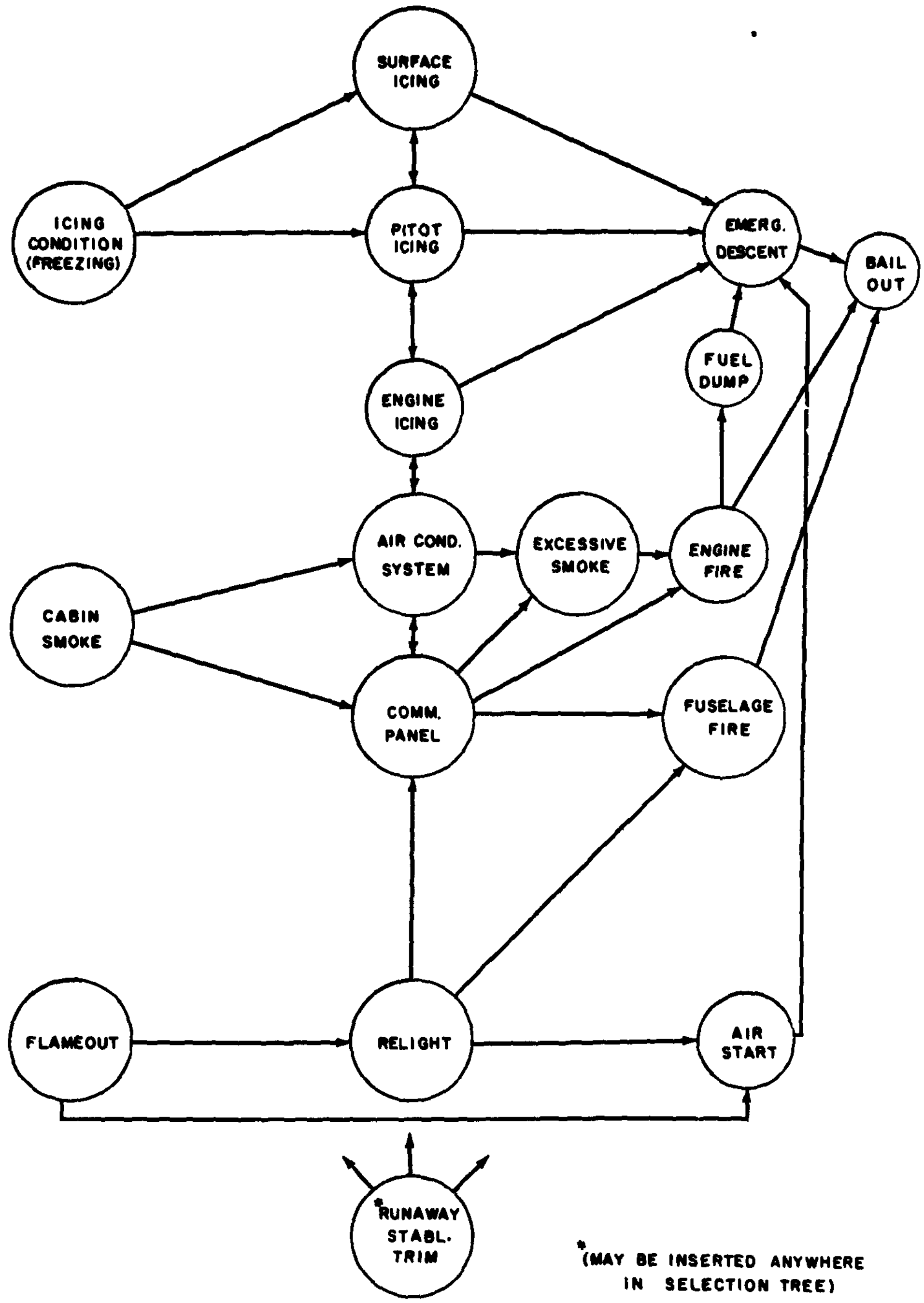


Figure 2 - Possible Task Sequencing in Operation-Teaching Mode

TABLE I Task Importance
(Rank Order and Percentage)

Task	Lt. A		Capt. B		Capt. C		Capt. D	Capt. E	Maj. F
	Rank	%	Rank	%	Rank	%	Rank	Rank	Rank
Engine Fire	4	90	5	88	8	49	1	4	6
Fuselage Fire	3	92	4	92	4	75	2	1	1
Engine Air Start	6	72	6	72	2	90	3	11	11
Bailout	1	100	1	100	7	50	4	2	5
Runaway Stabl. Trim	2	98	2	98	1	100	5	3	2
Emergency Descent	9	40	8	50	6	65	6	8	8
Smoke Comm. Panel	10	25	10	25	9	25	7	6	4
Engine Inlet Icing	11	20	-	-	-	-	8	9	-
Fuel Dumping	7	60	3	95	5	70	9	12	9
Smoke-Air Cond.	8	50	11	5	10	24	10	5	7
Smoke Elimination	12	1	9	40	11	1	11	7	3
Engine Relight	5	75	7	70	3	89	12	10	10
Pitot Icing									
Surface Icing									

TABLE II Task Difficulty and Complexity
(Rank Order and Percentage)

Task	Lt. A		Capt. B		Capt. C		Capt. D	Capt. E	Maj. F
	Rank	%	Rank	%	Rank	%	Rank	Rank	Rank
Engine Inlet Icing	8	20	-	-	-	-	1	9	-
Emergency Descent	6	49	8	50	2	70	2	8	7
Smoke Elimination	12	1	7	60	11	1	3	7	2
Engine Fire	4	60	5	85	7	50	4	5	6
Smoke Comm. Panel	10	5	3	95	9	25	5	6	4
Runaway Stabl. Trim	3	75	2	98	5	63	6	2	3
Engine Air Start	5	50	4	94	4	64	7	11	11
Bailout	2	85	9	30	1	100	8	3	5
Fuel Dumping	9	10	6	65	8	45	9	12	8
Fuselage Fire	7	30	1	100	6	60	10	1	1
Smoke-Air Cond.	1	100	10	25	3	65	11	4	9
Engine Relight	11	5	11	10	10	5	12	10	10
Pitot Icing									
Surface Icing									

Notes:

1. Task difficulty is a measure of how hard or complex the task is, or how long it takes (on the average) to master the task. Task Importance, as the name suggests, is a measure of the importance of the particular task in successfully operating the vehicle. There is not necessarily any direct connection between the two.

2. There appears to be little correlation between the rank ordering of task importance by the flight instructors. One reason for the disparity in individual rankings may be that they had trouble with that task either during their own training or during actual flight experiences.

3. Pitot Icing and Surface Icing tasks were not evaluated.

TASK SCORING CHART:

Instructor Action	Student's Score	Student's Response						
Icing Condition	%	Ignition sw. on before ice	Anti-ice on before ice	Note Instruments	Ignition sw. on	Retard Throttles	Eng. Anti-ice on	Circuit breaker check if icing persists
	100	1	2					
	95			1	2	3	4	5
	85			1	2	-	3	4
	65			1	-	2	3	4
	25			1	-	-	2	-
Importance of Steps		1	2	3	4	6	5	-

Notes:
1. 100% - This procedure prevents icing.
2. 95% - This procedure will counteract ice.
3. 85% - This procedure can be used during normal power.
4. 65% - This procedure will cause a compressor stall at high power.
5. 25% - This procedure will cause a flameout.
6. Instruments consist of RPM, exhaust pressure ratio, EGT.

FLOW DIAGRAM:

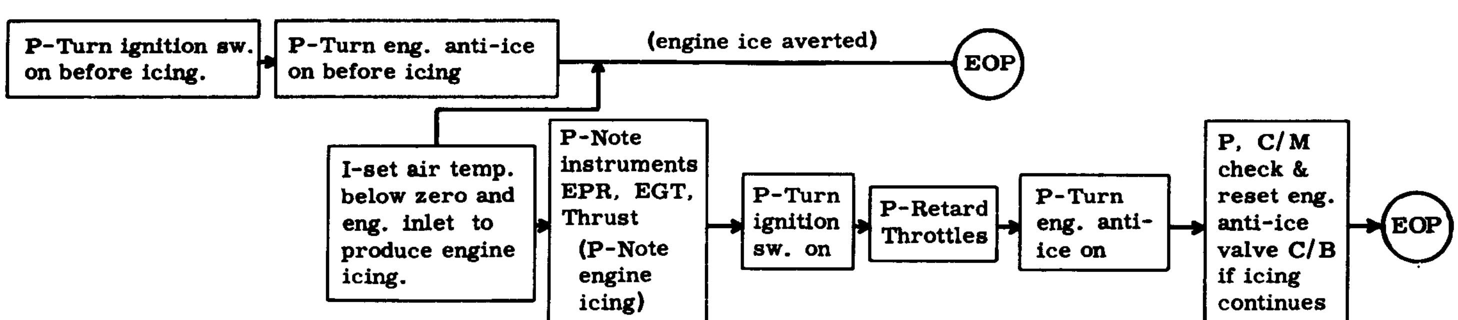


Figure 3 - Engine Inlet Icing (5 variations)

TASK SCORING CHART:

Instructor Action	Student Score	Student's Response						
Simulate icing fail pitot heat	%	Pitot heat on before icing	Note instr. malfunction	Put pitot on	Instrument Check	Fly out of ice area	Circuit breaker Check	Transfer Control
	100	1	2	-	-	3	-	-
	90	-	1	2	3	4	5	6
	70	-	1	2	3	-	4	-
	60	-	1	2	3	4	-	-
	30	-	1	2	-	-	-	-
	10	-	1	-	-	2	-	-
Importance of Steps		1	2	3	5	4	6	7

Note:
1. 100% would have averted an icing condition.
2. It is assumed that the pilot and co-pilot each have individual pitot systems, i.e., each system can be failed separately.
3. Pilots instruments only.

FLOW DIAGRAM:

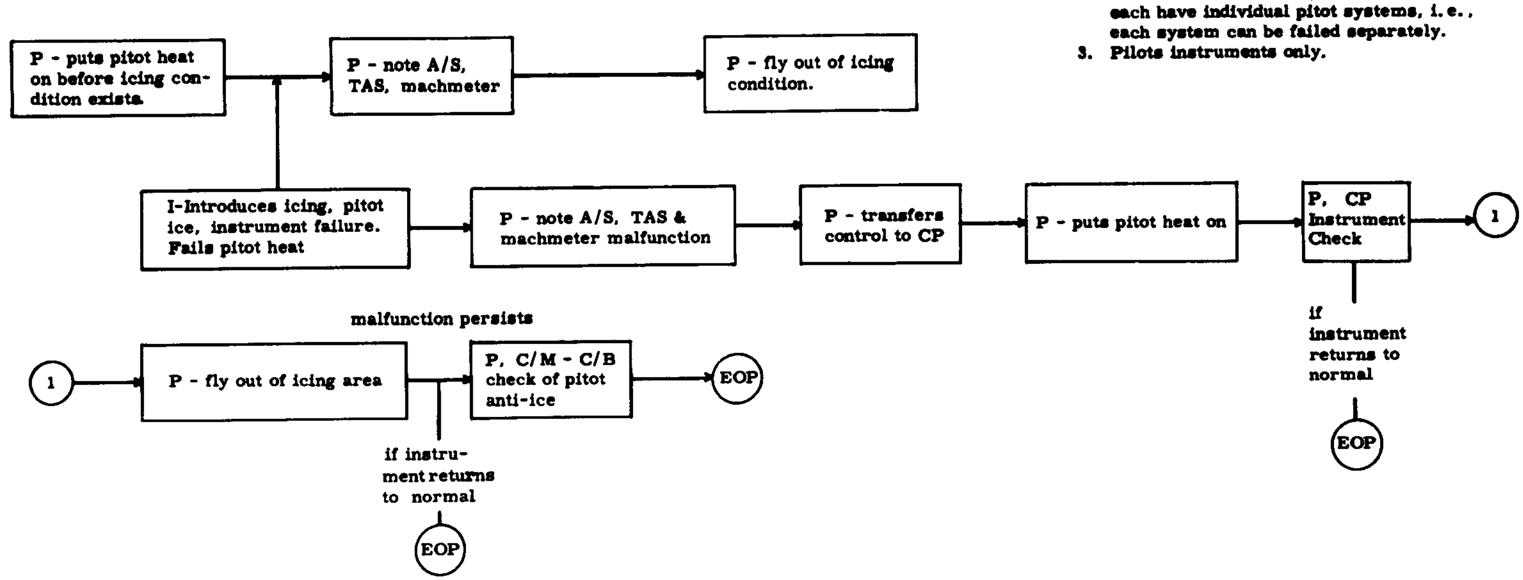


Figure 4 - Instrument Failure Due to Pitot Icing (6 variations)

TASK SCORING CHART:

Instructor Action	Student's Score	Student's Response			
Introduce Surface Icing	%	Put de-icing on before icing	Note wing & nose ice	Put de-icing on	Flies out of icing area
	100 60 40	1 - -	2 1 1	- 2 -	3 3 2
Importance of Steps		1	2	3	4

FLOW DIAGRAM:

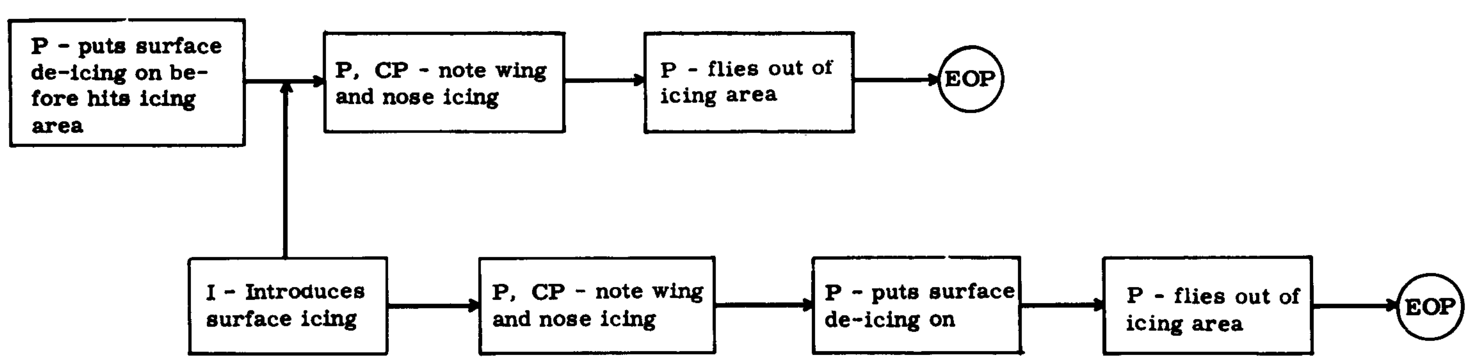


Figure 5 - Surface Icing (3 variations)

TASK SCORING CHART:

Instructor's Action	Student's Score	Student's Reaction				
Introduce smoke from air conditioner	%	Warns Crew	Uses 100% Oxygen	Reads Check List	Check Oil Pressure	Closes Bleed Sev.
	100	1	2	3	4	5
	93	2	1	3	4	5
	90	1	2	5	3	4
	45	1	2	-	3	4 (all)
	40	1	2	-	-	4 (all)
	25	1	2	-	3	4 (wrong one)
	10	-	1	-	2	3
	0	-	-	-	1	2
Importance of Steps		1	2	5	4	3

Notes:
1. 90% if forgot check list but later picked it up.
2. 45% if closed all the bleed switches including wrong ones.
3. 40% same as above but checked oil pressure.

FLOW DIAGRAM:

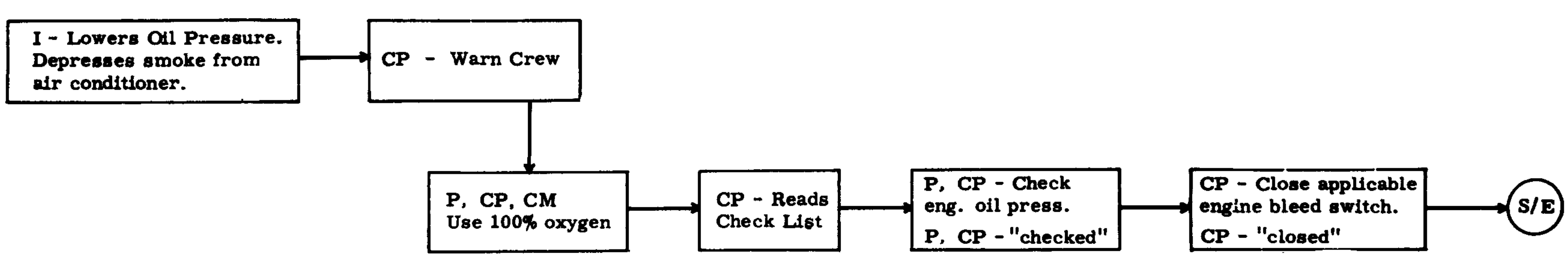


Figure 6 - Smoke From Air Conditioner (8 variations)

TASK SCORING CHART:

Instructor Action	Student's Score	Student's Reaction					
Initiates Smoke & Fumes	%	Sees Smoke	Warns Crew	Use 100% Oxygen	Reads Check List	Locates source	Pull out circuit Breaker
	100	1	2	3	4	5	6
	95	1	2	3	6	4	5
	90	1	2	3	-	4	5
	80	1	3	2	-	4	5
	40	1	2	3	-	4	-
	20	1	2	3	-	-	-
Importance of Steps		1	2	3	6	4	5

FLOW DIAGRAM:

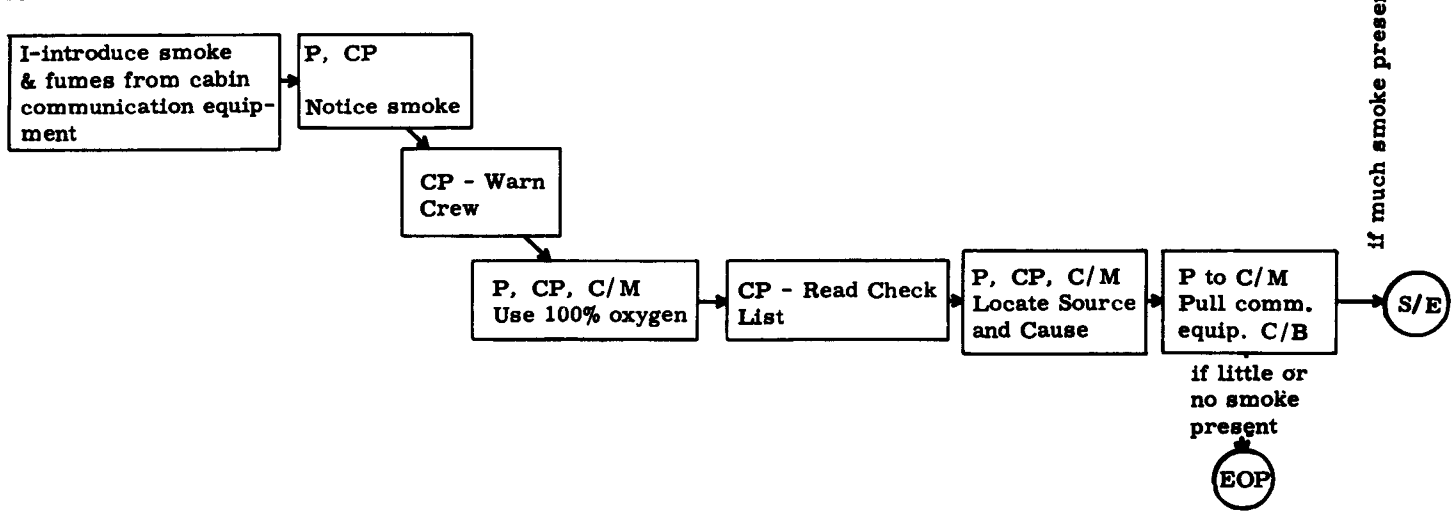


Figure 7 - Smoke and Fumes From Communications Panel (6 variations)

TASK SCORING CHART:

Instructor Action	Student's Score	Student's Response				
*	%	Read Check List	Close bleed Switches	Decrease Cabin Pressure	Set air cond. Ram Air	Return to Normal Oxygen
	100	1	2	3	4	5
	90	-	1	2	3	4
	85	-	-	1	2	3
	85	-	1	-	2	3
	80	-	1	2	3	-
	10	-	1	2	-	3
Importance of Steps		4	1	2	3	5

Note: * This procedure can be used as a follow on to smoke problems.

FLOW DIAGRAM:

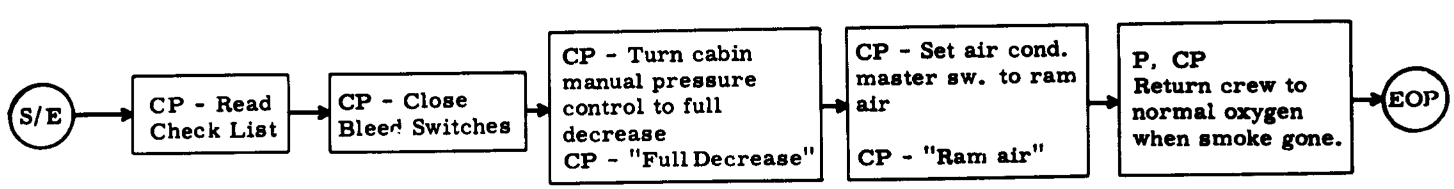


Figure 8 - Smoke and Fumes Elimination (6 variations)

TASK SCORING CHART:

(11 variations)

Instructor Action	Student's Score	Student's Response							
Push fuse-lage fire button	%	Warn Crew	Use Oxygen	Close bleed Switches	Cabin Pressure	Fight Fire	Use Check List	Ram Air	Repressurize
	100	1	2	3	4	5	6	7	8
	90	1	2	3	4	5	-	6	7
	80	2	1	3	4	5	6	7	8
	78	2	1	4	3	5	6	7	8
	75	1	2	3	4	5	6	-	7
	55	1	2	-	3	4	5	6	7
	50	1	2	-	-	3	4	5	6
	50	1	2	3	4	-	5	6	7
	45	1	2	3	4	-	5	6	-
	20	-	1	2	3	4	5	6	7
	0	1	-	2	3	4	5	6	7
Importance of Steps		1	2	3	4	5	6	7	8

FLOW DIAGRAM:

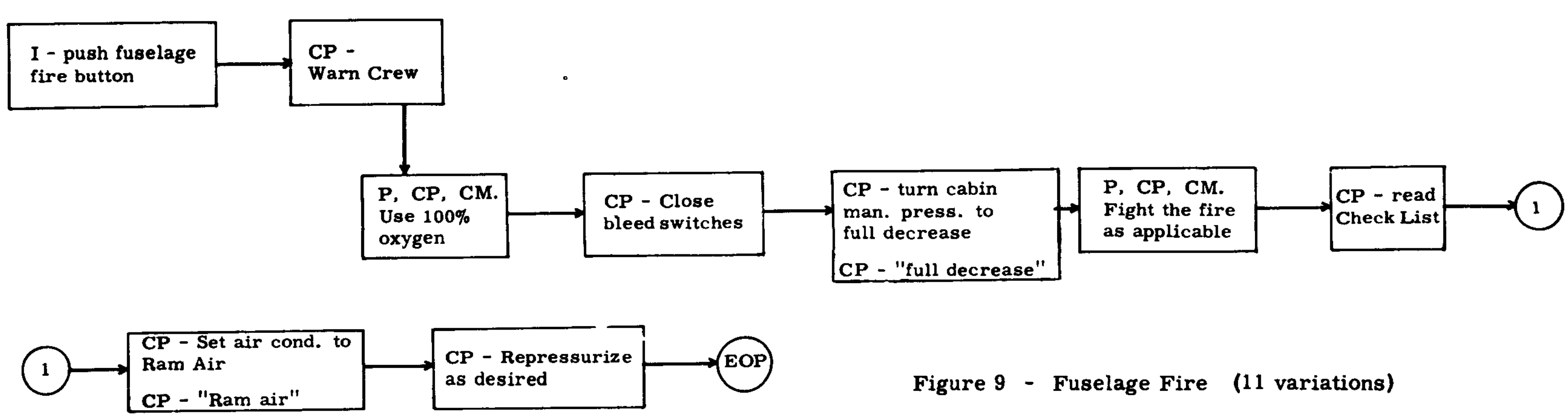


Figure 9 - Fuselage Fire (11 variations)

TASK SCORING CHART:

Instructor's Action	Student's Score	Student's Reaction					
Depress Eng. Fire	%	Check for fire	Throttle to Cut off	Pull fire switch	Read C/L	Eng. anti-ice off	fuel dump
	100	1	2	3	4	5	6
	98	1	3	2	4	5	6
	95	1	2	3	4	5	-
	95	1	2	3	-	4	5
	90	1	2	4	5	3	6
	90	1	2	3	4	-	5
	80	1	-	2	3	4	5
	60	1	2	-	3	4	5
	40	-	1	2	3	4	5
	0	1	-	-	-	2	3
Importance of Steps		1	3	2	6	4	5

Notes: 1. Fire warning indications may consist of: fire warning light or excessive EGT
2. All crew members check for fire.

FLOW DIAGRAM:

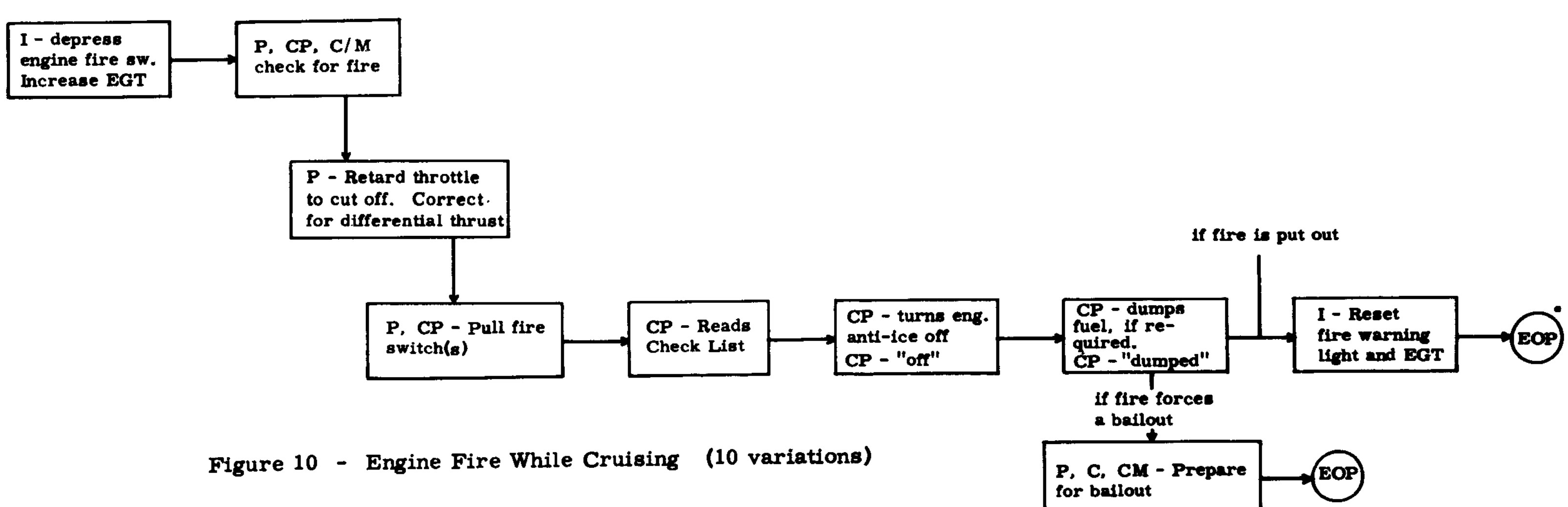


Figure 10 - Engine Fire While Cruising (10 variations)

TASK SCORING CHART:

Instructor's Action	Student's Score	Student's Reactions			
*	%	Close A/R to eng. man. valve	Open line valve	Actuate fuel dump. sw.	Actuate A/R pump switches
	100	1	2	3	4
	90	2	1	3	4
	90	1	2	4	3
	0	- (if open)	1	2	3
	0	1	-	2	3
	0	1	2	-	3
	0	1	2	3	-

* This procedure can be one of the steps in the engine fire procedure.

Note: 1. This procedure is more applicable to aircraft tankers than fighters and bombers.

FLOW DIAGRAM:

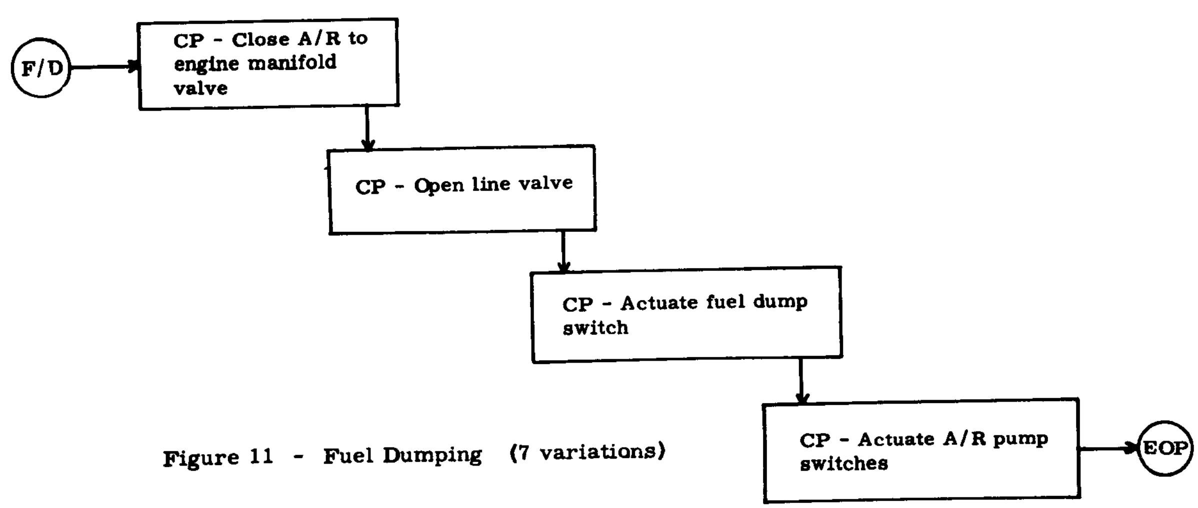


Figure 11 - Fuel Dumping (7 variations)

TASK SCORING CHART:

Instructor's Action	Student's Score	Student's Reaction			
Fail cabin pressurization *	%	Throttle to idle	Extend Landing Gear	Set Speed Brakes	Descent at max. speed via check list
	100	1	2	3	4
	95	1	3	2	4
	95	3	2	1	4
	95	2	1	3	4
	80	1	-	2	3
	80	1	2	-	3
	80	1	2	3	-
	70	-	1	2	3
	50	-	-	1	2

* This procedure can be part of an icing problem.

Note: 1. Reduce score one point for each second of delay between time cabin is depressurized and the descent procedure is started.

FLOW DIAGRAM:

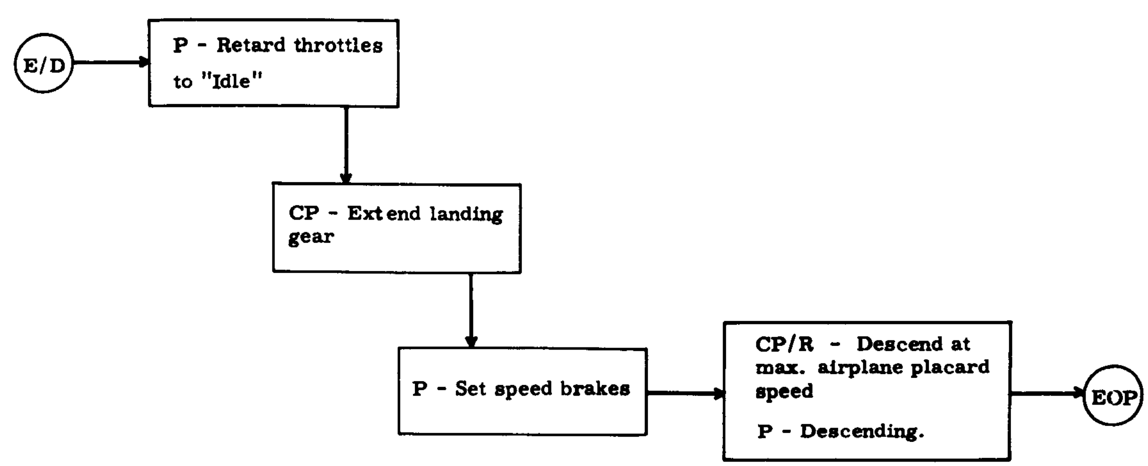


Figure 12 - Emergency Descent (9 variations)

TASK SCORING CHART:

Instructor's Action	Student's Score	Student's Reaction				
Bailout Button	%	Alert Crew	Put Oxygen Mask On	Release Cabin Pressure	Decrease Airspeed	Put Alarm Bell On
	100	1	2	3	4	5
	90	2	1	3	4	5
	40	2	1	3	-	4
	0	1	-	-	2	3
Importance of Steps		1	3	4	5	2

Notes:
1. The steps in this procedure are all very important.
2. Omission of a step will fail the student.
3. Any step taken after pushing the alarm bell does not count since the crew immediately bails out after alarm.
4. Decreasing airspeed consists of pulling back the throttle and using speed brakes.
5. The copilot reads the check list if time permits.
6. Bailout may be introduced by the instructor (bailout button) or as a result of progressive aircraft deterioration.

FLOW DIAGRAM:

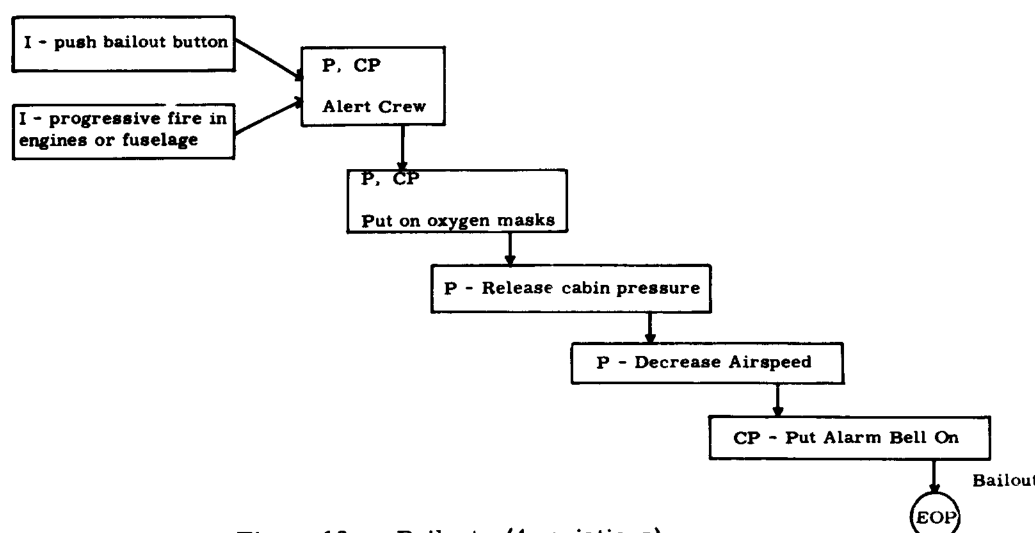


Figure 13 - Bailout (4 variations)

TASK SCORING CHART:

Instructor's Action	Student's Score	Student's Response					
Flameout EGT and fuel	%	See Engine Out	Retard Throttle	Set Starter	Use Check List	Check Fuel and EGT	Throttle to Cruise or C/O
	100	1	2	3	4	5	6
	90	1	2	3	-	4	5
	90	1	3	2	4	5	6
	80	1	3	2	-	4	5
	0	1	2	-	-	3	4
Importance of Steps		1	3	2	6	4	5

Notes:
1. The pilot must spot the flameout before significant RPM is lost.
2. If the starter is never set to Flight Start the engine will not relight.

FLOW DIAGRAM:

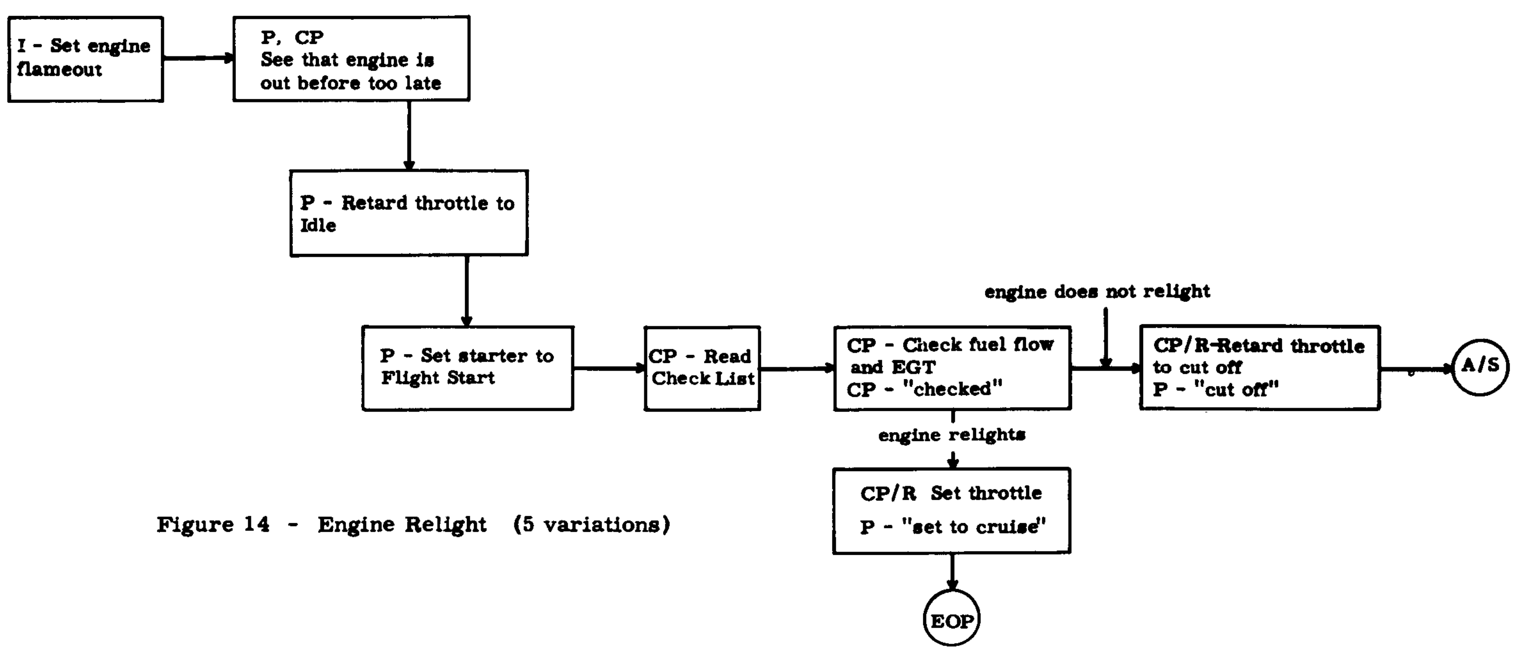


Figure 14 - Engine Relight (5 variations)

TASK SCORING CHART:

Instructor's Action	Student's Score	Student's Response											
Engine Failure	%	Use Check List	Throttle to Cut off	Fire Switch In	Boost Pumps On	Maintain Air-speed RPM	Check Oil Pressure	Starter to Flight Start	Throttle to Start	Instrument Check	Advance Throttle	Turn Starter Off	Reset Generator
	100	1	2	3	4	5	6	7	8	9	10	11	12
	98	1	3	2	4	5	6	7	8	9	10	11	12
	95	-	1	2	3	4	5	6	7	8	9	10	11
	95	1	2	3	4	5	6	7	8	9	10	--	11
	93	-	2	1	3	4	5	6	7	8	9	10	11
	85	1	2	3	-	4	5	6	7	8	9	10	11
	80	1	2	3	4	5	6	7	8	9	10	11	--
	45	1	2	3	4	-	-	5	6	7	8	9	10
	30	1	2	3	4	5	6	7	8*	9	10	11	12
	25	-	-	1	2	3	4	5	6	7	8	9	10
	10	1	2	-	3	4	5	6	7	8	9	10	11
	0	1	2	3	4	5	6	-	7	8	9	10	11
	0	1	2	3	4	5	6	7	8**	-	--	--	--
Importance of Steps		1	5	6	7	8	9	2	3	11	4	12	10

* Throttle to full
** Throttle to C/O

FLOW DIAGRAM:

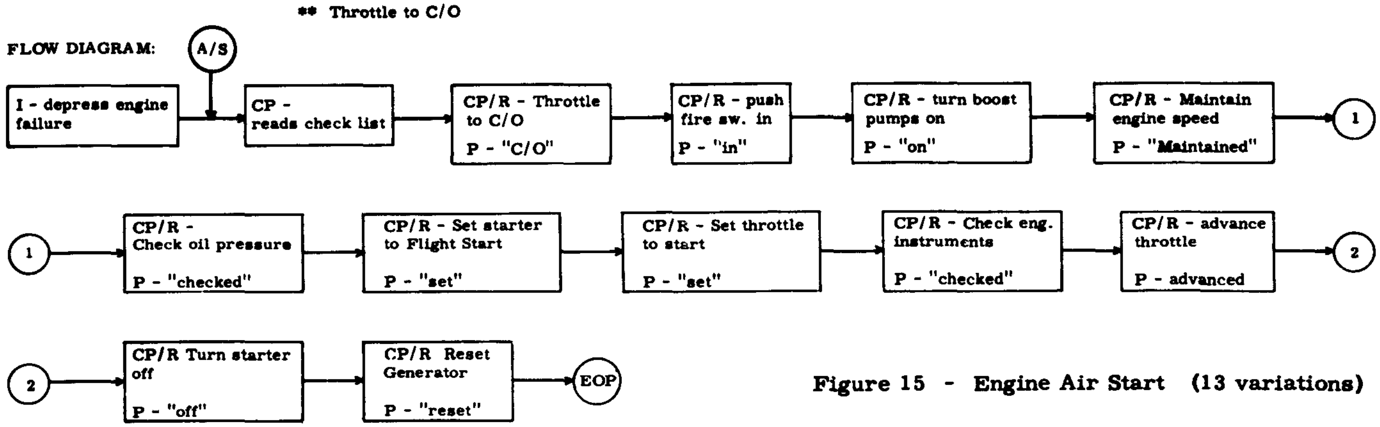


Figure 15 - Engine Air Start (13 variations)

TASK SCORING CHART:

Instructor's Action	Student's Score	Student's Response						
Runaway Stabl. Trim	%	Notice trim wheel & nose	Turn C/O switch	Stop trim wheel (manually)	Reverse trim control sw.	Check C/B	Manual Trim	Spoilers
	100	1	2^{NG}	3	-	4	5	-
	90	1	2^{NG}	3	-	-	4	-
	*90	1	2	-	-	-	3	-
	85	1	2^{NG}	-	3	-	4	-
	85	1	3	2	-	-	4	-
	85	1	3^{NG}	2	4^{NG}	-	5	-
	85	1	3^{NG}	4	2^{NG}	5	6	-
	85	1	2^{NG}	3	4^{NG}	6	-	5
	80	1	4	3	2^{NG}	-	-	-
	80	1	2^{NG}	4	3^{NG}	5	6	-
	70	1	4^{NG}	2	3^{NG}	5	6	-
	70	1	4	2	3^{NG}	-	5	-
	60	1	4^{NG}	3	2^{NG}	5	6	-
	60	1	4	3	2^{NG}	-	5	-
	40	1	4^{NG}	3	2^{NG}	-	-	6
	25	1	-	-	-	-	2	-
	10	1	-	-	-	-	-	2
	0	1	-	-	-	-	-	-
Importance of Steps		1	2	3	6	5	4	7

Notes:
1. NG indicates that switch was inoperative.
2. "Check C/B" - includes holding trim wheel to keep plane level while applicable circuit breakers are being checked and pulled.
3. 10% may generally be deducted for omitting the C/B check.
4. 15% may generally be deducted for substituting spoilers for manual trim.
5. In this task the same circuit breaker works both the pilot's and copilot's trim control switch.
6. Three stabilizer trim mechanisms may influence aircraft operation (1) manual trim control wheel, (2) trim control switch. (3) trim cutout switch.
7. The automatic pilot does not influence the student's score in this task.
8. *In this case 90% was the most the pilot could get.
9. Manual trim must be used for rest of the flight after the runaway is brought under control.

FLOW DIAGRAM:

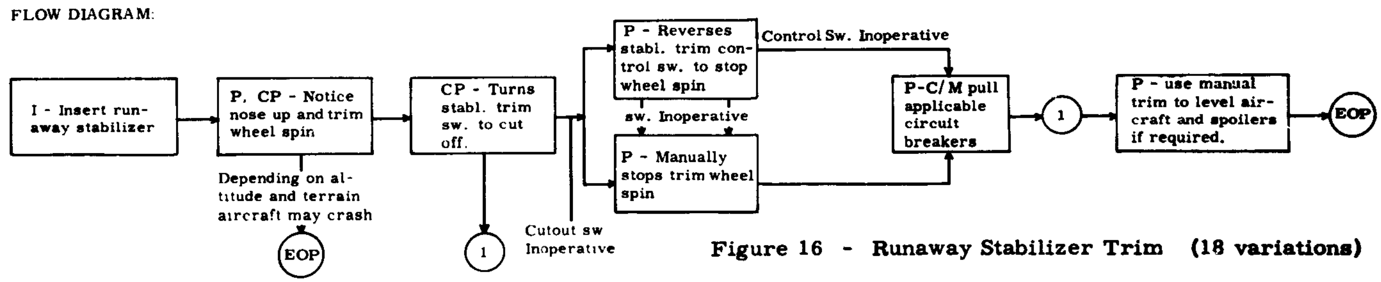


Figure 16 - Runaway Stabilizer Trim (18 variations)

SECTION III

ALGORITHMS

Two algorithms are now presented for sequencing tasks to the student. One is for teaching in-flight, emergency tasks associated with the operation of a flight vehicle, that is, the teaching of procedural operations. This is termed the operation-teaching mode of the Task-Sequencer. The other is termed the tactic-teaching mode and is directed toward teaching the strategy-of-operation of a flight vehicle to demonstrate to the student (crew) the implications of their decisions in response to hostile situations.

There are two sub-modes in the operation-teaching version of the Task-Sequencer. One is for single-task presentations per cycle where a cycle consists of the selection and presentation of the task, the student's response, and the performance evaluation. The other is for multitask presentations per cycle. Furthermore, in the multitask sub-mode, there are two variations for determining which tasks are to be presented.

The algorithm given for selection of tasks in the tactic-teaching application operates upon a pay-off, or value, matrix. A number of training or procedural models are appropriate with that mode, for example those already discussed in the section I, Applications of the Task-Sequencer.

A general schematic and discussion of the structure and information flow of a heuristic program are then presented in terms applicable to a number of models.

An operational definition and explanatory discussion of the algorithms are now given along with flow charts.

OPERATION-TEACHING MODE OF TASK-SEQUENCER

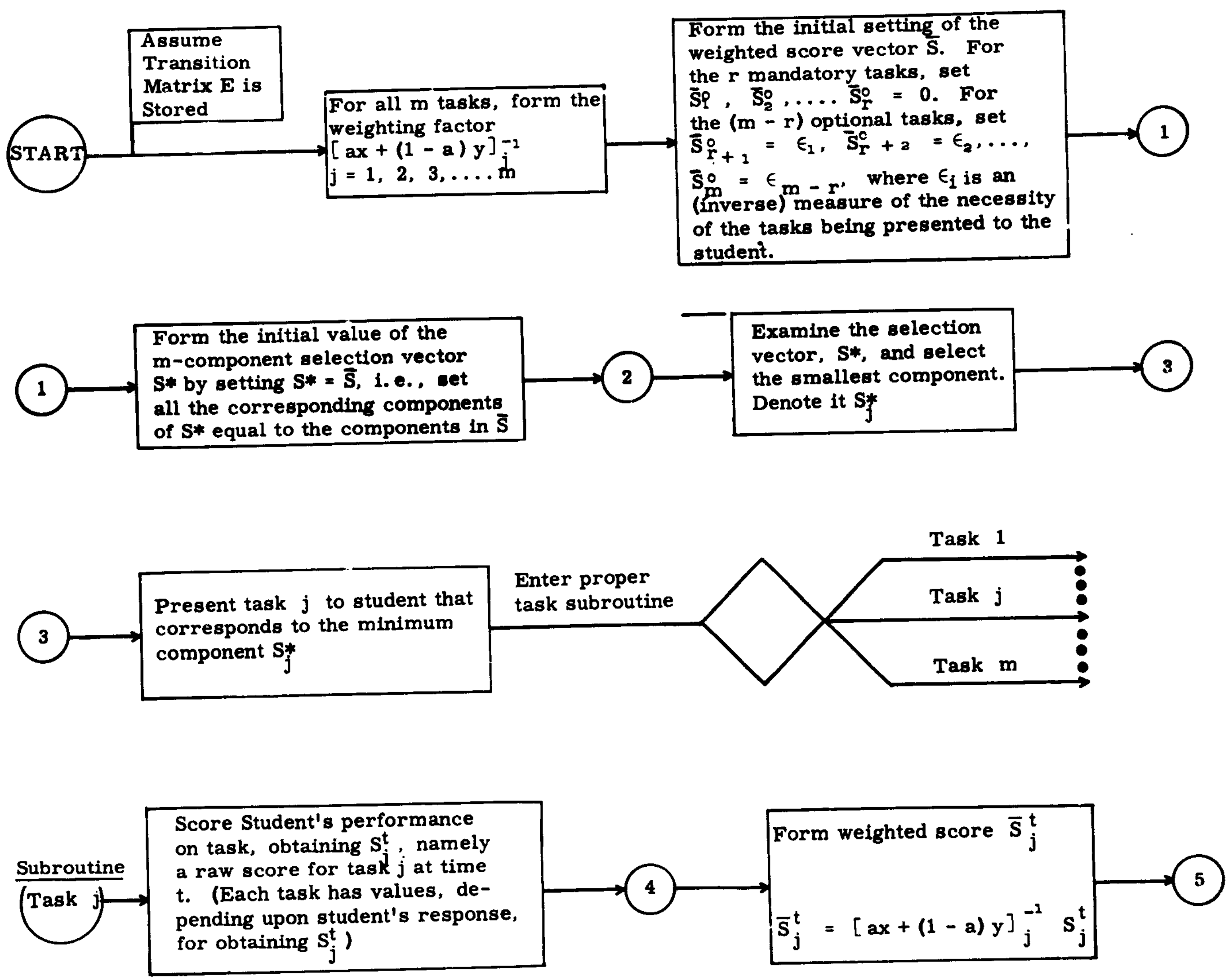
In this mode, the Task-Sequencer selects tasks to present to the student, so that (1) performance on all mandatory tasks is tested and (2) those tasks in which the student performed poorly, or in a non-optimal manner, are stressed. A mandatory task is one on which the student must be tested. On optional tasks, the student need not necessarily be tested on all of them -- only a sampling is desired. The better the student, the larger the number of optional tasks given.

The algorithm that has been formulated to satisfy these goals is now presented. See figure 17.

Single-Task Presentation

Each task has associated with it a couplet (x, y) where $0 < x, y \leq 1$, and in which x and y are measures respectively of difficulty and importance of the task that is to be mastered by the student. (See tables I and II, page 10, for an illustration.) Furthermore, for the set of tasks there is a proportionality constant, a, ($0 \leq a \leq 1$) relating the relative weight or influence of the task difficulty, x, and the task importance, y. A weighting factor, $ax + (1 - a)y$, is formed for each task. This is used to modify the raw score obtained by the student from his performance on specific tasks. The specific value of the proportionality constant, a, really a value judgement, depends upon the instructor's estimation of the skill of each student pilot.

For example, in the case of a student who is a relative novice, the instructor would specify a value of zero (or close to zero) for a, thereby causing x, the measure of difficulty, to vanish from the expression. This would have the effect of first focusing attention on those tasks which have a high importance value, and are suggestive of intensive training, to emphasize their being learned. Conversely, for a well trained student, the instructor may wish to emphasize those tasks which are deemed most difficult by setting the value of a close to, or equal to, one.


START
Assume Transition Matrix E is Stored
For all m tasks, form the weighting factor $[ax + (1 - a)y]_j^{-1}$ $j = 1, 2, 3, \ldots m$
Form the initial setting of the weighted score vector $\bar{S}$. For the r mandatory tasks, set $\bar{S}_1^o, \bar{S}_2^o, \ldots \bar{S}_r^o = 0$. For the (m - r) optional tasks, set $\bar{S}_{r+1}^o = \epsilon_1, \bar{S}_{r+2}^o = \epsilon_2, \ldots, \bar{S}_m^o = \epsilon_{m-r}$, where ϵ_i is an (inverse) measure of the necessity of the tasks being presented to the student.
1
1
Form the initial value of the m-component selection vector S* by setting $S^* = \bar{S}$, i.e., set all the corresponding components of S* equal to the components in $\bar{S}$
2
Examine the selection vector, S*, and select the smallest component. Denote it S_j^*
3
3
Present task j to student that corresponds to the minimum component S_j^*
Enter proper task subroutine
Task 1
Task j
Task m
Subroutine
Task j
Score Student's performance on task, obtaining S_j^t, namely a raw score for task j at time t. (Each task has values, depending upon student's response, for obtaining S_j^t)
4
Form weighted score $\bar{S}_j^t$
$\bar{S}_j^t = [ax + (1 - a)y]_j^{-1} S_j^t$
5

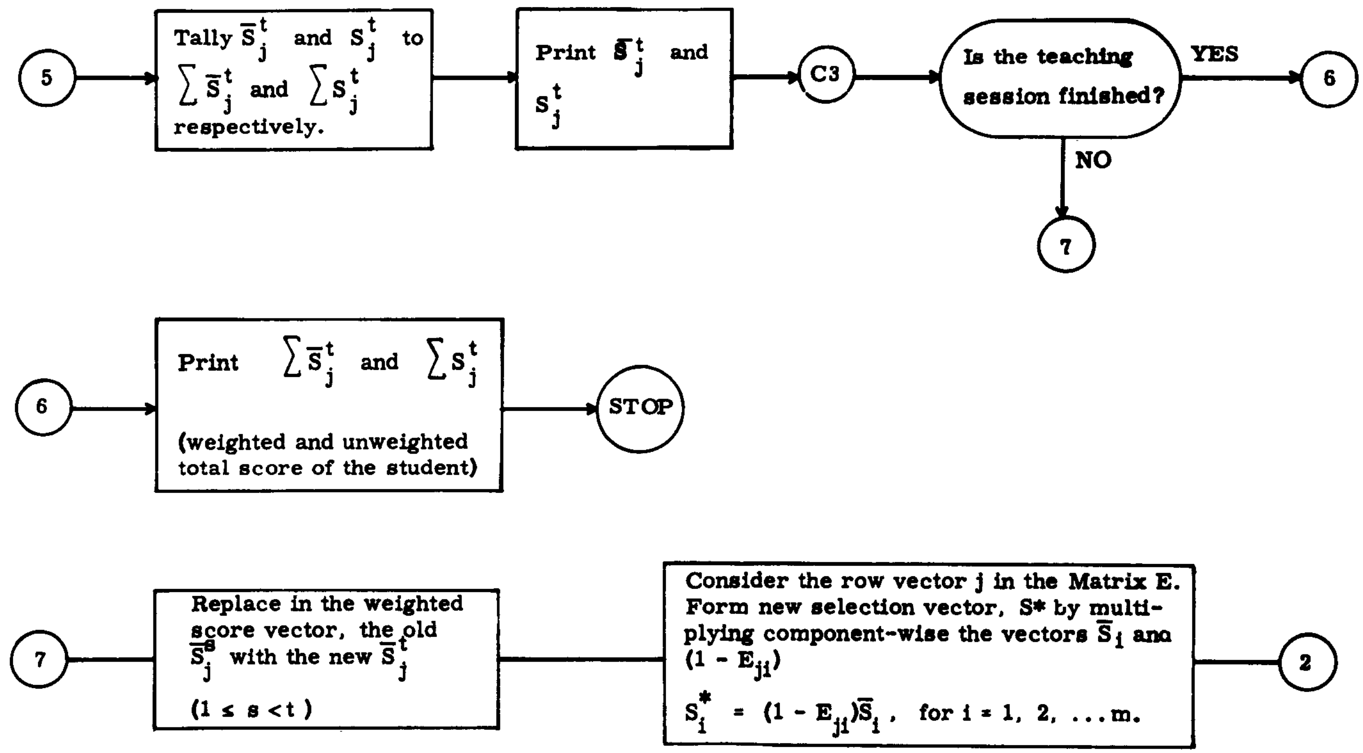


Figure 17 - Flow Chart for Operation-Teaching Mode of Task-Sequencer

(Clearly, there is no necessary correlation between the difficulty of a task and its importance. It is possible for a task to be both difficult and important, and thus emphasized for $\underline{a}$ equal to one or zero.) Since $\underline{a}$ ranges between zero and one, any mix of emphasis on difficulty and importance can be obtained. Thus, with this simple mechanism it is possible to take into account the various abilities of individual students and adjust the sequencing of tasks correspondingly.

During the training session the performance of the student on a task gives a raw score S_j^t, for task j at time t. Each task has predetermined values for obtaining the raw score. This raw score is then converted to a weighted score, $\bar{S}_j^t$, by multiplying the reciprocal of the weighting factor and the raw score.

This process, continued in time over many cycles, yields a vector of weighted scores (see figure 18a) with each entry representing the last performance score recorded, or if desired, some average of the scores, for the respective tasks. Thus for m tasks there are m entries. Each entry is computed at different points in time. This weighted vector forms the nucleus of a task selection scheme.

	Tasks	Parameters	Weight Factor	Weighted Score	Selection Vector $[S_i^* = (1-E_{ki})\bar{S}_i]$ (i = 1, 2, ... m) (k = number of last task selected)
Mandatory Tasks	1	$(x, y)_1$	$[ax + (1-a)y]_1^{1}$	$\bar{S}_1^{t-1}$	S_1^*
	⋮				⋮
	r	$(x, y)_r$	$[ax + (1-a)y]_r^{-1}$	$\bar{S}_r^t$	S_r^*
Optional Tasks	r + 1	$(x, y)_{r+1}$	$[ax + (1-a)y]_{r+1}^{-1}$	$\bar{S}_{r+1}^{t-3}$	S_{r+1}^*
	⋮				⋮
	m	$(x, y)_m$	$[ax + (1-a)y]_m^{-1}$	$\bar{S}_m^{t-4}$	S_m^*

Figure 18a - Data Vectors

		1	2	. . .	m
	1				
	2				
	⋮				
last task selected →	k	E_{k1}	E_{k2}	. . .	E_{km}
	⋮				
	m				

Figure 18b - Transition Matrix E

Figure 18 - Information for Operation-Teaching Mode

As the algorithm is presently described, the next task chosen will be based on the smallest entry in the weighted vector. Initially, for an unbiased task selection, all the tasks can have their representative entry in the weighted vector equal to zero. Note, there is a natural selection of the untested tasks first for initially their entries are equal to zero, the smallest permissible value, and thus no separate initial selection rule is necessary. As time progresses these zero entries are replaced by an actual performance score.

The process described above is designed for a list of mandatory tasks, all of which must be tested. This appears most useful at present. The technique however may be easily generalized to include both mandatory and optimal tasks. This could be simply accomplished by initially setting only the mandatory task entries in the weighted score vector equal to zero; the optional tasks would initially be set with some positive value. This provides a certain balance of testing and teaching time for both sets of tasks, while still guaranteeing complete testing of the mandatory tasks.

However, another matter of interest is that some tasks virtually demand, for intelligent and meaningful task sequencing, that only a restricted sub-class of the total number of tasks follows immediately. For example, very often in an actual flight a fire in one section of the plane is followed by trouble in the ventilation system or by trouble in the electrical system.

This could be handled by having a set of "permissive sentinels" associated with the tasks. Those tasks that require a restricted class of subsequent problems would activate the proper permissive sentinels, namely, those in the restricted class. Only those tasks that have their permissive sentinels activated would then be considered.

Another solution, although more complicated and hence more time consuming, appears much better than the above-mentioned naive approach. Namely, each pair of tasks is assigned weights according to the probability of selection of the second task in the pair after a choice of the first task. As a practical example, trouble in the ventilation system is more likely to occur after an engine-fire than after trouble with the electrical system -- though both eventualities may occur.

More formally, consider a transition matrix E to be stored in the computer (see figure 18b). This transition matrix is of order m for a simulation consisting of m tasks, in which the entry E_{ij} denotes the probability (relative frequency of occurrence) of task j being selected if the system is in state i (task i). Of course, each row of matrix E sums to unity.

Instead of choosing tasks on the basis of the weighted score vector, $\bar{S}$, a slightly different scheme is now used. A selection vector S^* is generated and used for that purpose. This is done in the following manner. Assume that task k was selected for the past task cycle. The row vector k in the matrix E is now considered, and a new vector consisting of the one's complement of the entries of the vector k is formed, namely $(1-E_{ki})$, for $i = 1, 2, 3, \ldots, m$.

This vector $(1-E_{ki})$ provides another weighting factor for the weighted score vector $\bar{S}$, namely $\bar{S}$ and $(1-E_k)$ are multiplied component-wise to form S^*.

$$S_i^* = (1-E_{ki})\bar{S}_i \,, \quad \text{for } i = 1, 2, \ldots, m$$

The task associated with the smallest entry in S^* is chosen next for presentation. The one's complement of row k of the E matrix, $(1-E_k)$, is used as a weight for $\bar{S}$, for the larger the value of E_{ki}, (by definition) the greater the frequency that task i should follow task k. Thus since the smallest entry in S^* determines the choice of the next task, the mapping $(1-E_k)$ is employed. Figure 17 gives this scheme.

If there is zero probability that task i will follow as a direct result of task k, then the coefficient for the corresponding ith entry in S^* will be equal to 1, the largest possible coefficient. Since the task associated with the minimum entry of S^* is selected next, increasing the value of the coefficient decreases the probability of an immediate selection of that task. Of course it is possible that task i will be chosen next because of $\bar{S}_i$ being extremely small. That corresponds to the physical interpretation that two independent events or malfunctions occur. Initially S^* is set equal to $\bar{S}$.

Another technique that was examined for selection of the tasks consists of forming and storing the difference of two successive values of the entries in the selection vector. The last set of entries in the selection vector would be scanned and then, of all those entries below a certain threshold, the minimum difference would be selected. This corresponds to a "derivative" or slope -of-improvement. Clearly a threshold is necessary to select the proper subset of "derivatives"; if a student completely masters a certain task, then by definition, he cannot demonstrate any improvement on new trials. Reselection of that task would be a waste of training time.

The difficulty with this technique along with other variants such as examining the second or third order differences, corresponding to "acceleration" and "jerk" (the rate of change of acceleration) is that too much statistical evidence would have to be gathered. Even if one wished to keep records of each student from one time of the year to the next (from one training period to the next) a false bias might cause an ill-choice of tasks. It appears to be analogous to too heavy smoothing. However, after the Task-Sequencer is mechanized and in operation, one may wish to experiment with these selection-rules.

If desired, emphasis could be given to the response time of the student in the computation of S_j^t, the raw score for task j at time t. As presently conceived, the student is allowed a certain amount of time to respond. If he exceeds the time limit he defaults and receives a score of zero, regardless of his performance. A minor variation would be for each task to have an individual time limit rather than the same time limit for all tasks. Actually there can be a manual override to eliminate totally the matter of the time constraint which would be under control of the instructor. In general, however, it would not be desirable to eliminate the time factor.

Another scoring variant is to give "bonus points" if a correct response is made within a certain time tolerance. All this would require is the inclusion of lower, as well as upper, thresholds for time-of-performance.

Multitask Presentation

It is desirable to include provisions in the operation-teaching mode for multitask presentation. For advanced or superior students, who are well versed in handling any one single-task, it is possible to increase the value of a training session by this device. For multitask presentation there are two possible variations.

The first is dependent and involves presenting the student with two or more tasks that are strongly conditional upon one another. Essentially this involves a compression-in-time. In an actual flight, one emergency will lead to others unless corrected at once. Presenting these emergencies to the student at once, i.e., in one time cycle rather than spread out over many time cycles, is equivalent to demanding remedial action in a situation that has suddenly deteriorated, perhaps initially through pilot/copilot carelessness.

The second is independent and does not attach any special significance to the conditional dependency of the various tasks in the multipresentation of them. This is equivalent to the student operating a badly degraded flight vehicle that is subject to a number of malfunctions not necessarily due to any initial carelessness on the part of the student. This stress laden situation would provide a fairly effective test of the student's skill.

Which variation is optimal can only be determined by some empirical investigations undertaken after the Task-Sequencer has been mechanized and is operational. What is most probable is that both would be useful to have at the disposal of the instructor.

The simplest manner to implement the first variation would be to choose the primary task by the same technique used in single-task selection; i.e., via the selection vector. The row in the transition matrix that corresponds to the choosen task is then examined for the largest entry. The column number associated with this entry selects another task to be scheduled for presentation as one of the multitasks. The row in the transition corresponding to the newly selected task is now examined for its largest entry to determine an additional task. This procedure continues until (g-1) new tasks have been selected for presentation along with the primary task -- the one that started the task-chain. A test would be made to insure that there are no replications in the (g-1) tasks. If a repeat of a task already scheduled for presentation in that (scoring) cycle is threatened, then the entry corresponding to it is ignored and the next largest entry in the row is chosen.

Actually this amounts to taking minors of the transition matrix by temporarily (for that cycle) deleting the column and row corresponding to a selected entry for the choice of another entry.

In lieu of this it might seem that the optimum procedure to follow would be to square the original transition matrix, and then consider the row that corresponds to the primary task, and select the largest element in that row. Then cube the original matrix, and select the largest element in the row, and so forth. The motivation for that would be the simple fact that raising a transition matrix, E, to the gth power gives entries E^g_{ij} representing the probability of going from state (task) i to state (task) j in exactly g steps. However it does not give a chain of tasks which is what we are interested in obtaining. Furthermore, one entry in the gth power of the transition matrix might (in fact, generally will) acquire magnitude from a number of task-chains. In reality only one task-chain is permitted for one time cycle. Thus, that technique definitely should not be followed.

The second variation (representing the badly degraded flight vehicle) would be implemented by choosing g smallest entries in the selection-vector, for g multitask presentations, in one cycle. After the evaluation of the performance of the student on these g tasks, the modification of the weighted score vector and the selection vector would be accomplished. The modification would proceed by considering each task individually and treating it as in the single-task selection. In effect it would be the same as if the g tasks were presented over g task cycles, with the g tasks chosen in advance at the start of the first task cycle. Naturally only one task cycle, not g cycles, will be employed for these g tasks; therefore, an extra amount of time may be allocated in that task cycle for performance of these tasks.

Flow charts, figure 19, pages 32 and 33, are now given for the two multitask variations discussed in the preceding paragraphs. As one would suspect, the flow chart for multitask presentation employs most of the same routines as the flow chart for single-task, plus some additional control routines. Rather than duplicate the same flow chart as in the single-task submode, figure 17, the additions are now given in flow chart form, figure 19, and the places where these new routines are to be incorporated are indicated in the subsequent discussion.

The entrance connector 3A of the multitask flow chart, figure 19, is to be placed, or inserted, directly before connector 3 of figure 17. The terminal connector C3 in this new addition to the flow chart is to be placed directly before the interrogation "Is the teaching session finished?" shown in figure 17.

As indicated in the box immediately following connector C2 a portion of the original flow is to be employed as part of the multitask sub-routine, namely, the section shown in figure 17 from task j to the interrogation, "Is the teaching session finished?"

Whether or not it would pay to have a separate program for multitask presentation or to incorporate the submodes in one routine to be held in memory at one time can be answered on the coding level. Essentially, the question boils down to the amount of memory available and the speed of the program input devices.

Estimate of Memory Space

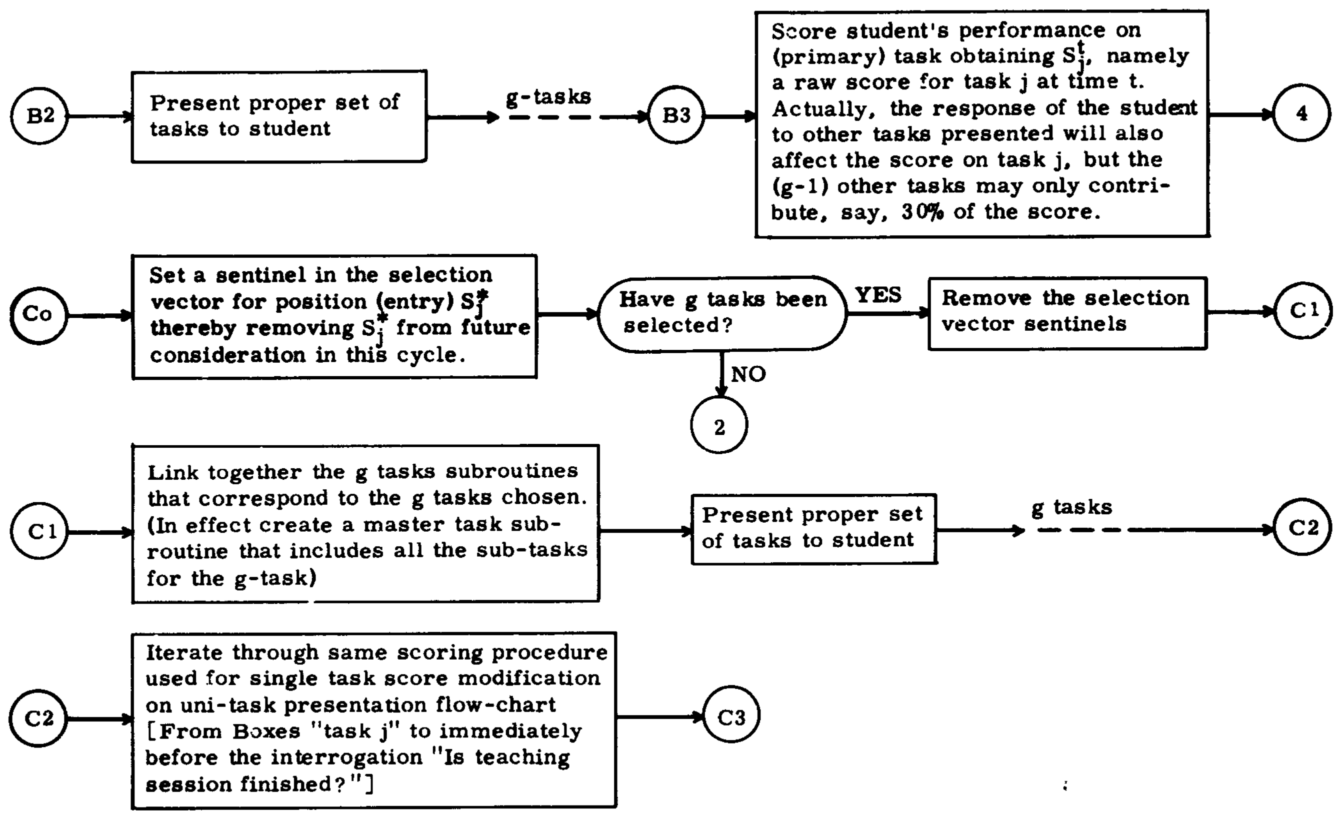
Some rough memory estimates are given for the operation-teaching mode of the Task-Sequencer. The purpose of these estimates, necessarily only approximations or "ball-park figures," is to provide some basis for determining the number of memory units to be ordered for a computer system.

The amount of memory space required is a direct function of the number of tasks that are to be used for the teaching session. Accordingly, the evaluation was made with the number of tasks, denoted by m, as a variable.

An estimate of memory space required is given by the quadratic function:

$$m^2 + 45m + 100$$

The term m^2 is given by the contribution of the transition matrix, E, which is of order m. Five vectors of length m must be stored (two for the measures of the task's importance and difficulty,


B2
Present proper set of tasks to student
g-tasks
B3
Score student's performance on (primary) task obtaining S_j^t, namely a raw score for task j at time t. Actually, the response of the student to other tasks presented will also affect the score on task j, but the (g-1) other tasks may only contribute, say, 30% of the score.
4
Co
Set a sentinel in the selection vector for position (entry) S_j^* thereby removing S_j^* from future consideration in this cycle.
Have g tasks been selected?
YES
NO
2
Remove the selection vector sentinels
C1
C1
Link together the g tasks subroutines that correspond to the g tasks chosen. (In effect create a master task subroutine that includes all the sub-tasks for the g-task)
Present proper set of tasks to student
g tasks
C2
C2
Iterate through same scoring procedure used for single task score modification on uni-task presentation flow-chart [From Boxes "task j" to immediately before the interrogation "Is teaching session finished?"]
C3

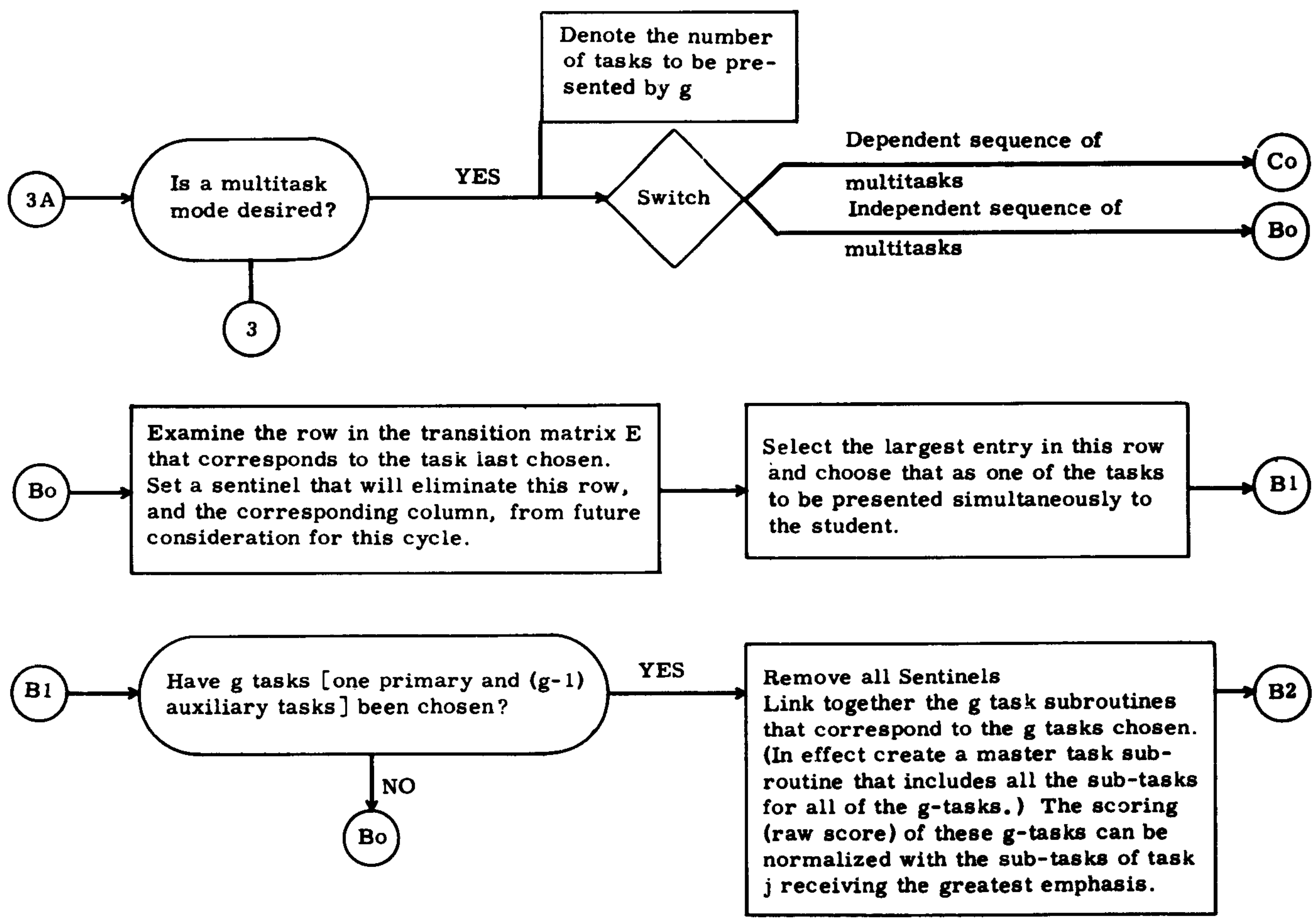


Figure 19 - Multitask Presentation

one for the weight factor, one for the weighted score, and one for the selection vector), and upon examination of some typical flight tasks, it appears that 40 instructions should be sufficient for each task. Thus the term 45 m is obtained. Finally, upon close examination of the controls of the routine, an estimate is made that not more than 100 words are necessary. The sum of these terms yields the quadratic given above.

Thus, for thirty tasks, i.e., m = 30, the amount of memory is 2,350 words.

It must be repeated that this estimate is for the purpose of supplying a "ball-park figure." However, it is believed that the estimate is fairly realistic -- perhaps on the conservative side. No doubt, when the algorithm is actually coded in a specific computer, advantage can be taken of various special features.

Of course the other side of the coin is that new features and further sophistications may be added later on. It is a fairly good rule that systems always increase in complexity in time. Thus the more memory available, the more flexibility and growth potential.

TACTIC-TEACHING MODE

As previously noted, the student-instructor relation, in the tactic-teaching mode, can be cast into the form of a zero-sum game. Hence the mathematical approaches suggested by game theory are appropriate. This approach stresses the development of student experience in choosing various strategies of operation or alternate goals in the face of active enemy opposition.

At any moment in time (or equivalently, at any one move) the choice of moves, or tactics, and their resulting value, may be represented by a "pay-off matrix." The columns and rows of this matrix represent the choices of tactics open to the student and the Task-Sequencer, respectively. Various models for this mode of the Task-Sequencer have been discussed in previous sections.

It is recognized that the entries in the pay-off matrix need not be, and in general will not be, a single value. A density function for each entry would provide more realism. The distributions, however, could be replaced by some random variable such as the expectation, or even by some conservative or pessimistic estimate.

Furthermore, it is recognized that these distributions are nonstationary. The game is of a dynamic nature, namely, the distributions of their estimates may change in time as a function of the history of the game. In the extreme, the pay-off matrix may even gain new columns and/or rows, and lose old ones. This gain-or-loss of rows and columns corresponds to the addition or deletion of the various tactics open to the Task-Sequencer and the student.

The student need not necessarily know the specific pay-off matrix to play-the-game. In fact, in a real-life situation he might have only crude ideas of the expected pay-offs for his actions. The expected value for his possible actions is something the student would find out for himself by a combination of empirical investigation (experimenting) and pre-study of the situation.

Heuristics operating upon the last specific position or choice of the Task-Sequencer and the student, i.e., the last situation, might provide the best means of modifying the distributions during the course of a game in the dynamic situation. This area requires considerable study.

As an initial approximation to the full problem, it is assumed that the order of the matrix and all of its entries are static. An algorithm is now presented for the Task-Sequencer that is based upon the game-theoretic mathematical work of G. W. Brown (ref. 2) and J. Robinson (ref. 3). It is a "learning" technique in that the choice of play on the Task-Sequencer's part is derived from the past history of results. It has the virtue of being computationally simple, and thus can conform to the real-time requirements of the Task-Sequencer. The flow chart of the algorithm is given in figure 20.

Consider a pay-off matrix in which the rows denote the strategies available to the Task-Sequencer, and the columns denote the responses open to the student.

The first choice, or move, of the Task-Sequencer is made by the following process: Each row in the pay-off matrix is summed, thereby giving for an $m \times n$ matrix, m sums. The row is chosen which has the largest sum. If two or more items are tied for the largest sum, the sum of the squares of the entries is formed for these candidate rows. The tie is then resolved by selecting the row in the pay-off matrix associated with the smallest value given by the new measure. If there are still ties, then a choice is arbitrarily made.

The summing of the entries corresponds to computing the expectation for each row, under the Laplacian assumption that with ignorance of any previous history of choice or preference by the student, each n column should be tentatively assigned the same probability of being selected, namely, $1/n$. The reward or pay-off is, of course, given by the values of the entries. Since all columns have the same number of entries, there is no need to multiply each entry by the same $1/n$ constant; summing is sufficient.

The tie-breaking scheme corresponds to computing the variance (spread) of the entries, and selecting the row exhibiting the smallest deviation from its expected value (a conservative rule). Since the expected value is the same for all of the candidate rows, there is no need to subtract the square of it from the newly computed row-measures.

After the first move or play on the part of both sides, the choice of moves by the Task-Sequencer proceeds in a recursive fashion by constructing a cumulative vector S by adding the entries of the column vector chosen by the student at each of his moves to the corresponding entries of S.

To select his tactics for the next move or play, the Task-Sequencer scans the cumulative vector and notes the position of the largest entry, assuming that the Task-Sequencer wishes to maximize the pay-off. He then plays the move that corresponds to this entry; e.g., if the largest entry is in the kth position of the cumulative vector, the Task-Sequencer selects the kth row for his move. The process is then duplicated for the next play and so on. Ties may be resolved by the tie-breaking scheme adopted in the initial selection by the Task-Sequencer.

One may take advantage of an a priori estimation of what the student is most likely to do -- perhaps based on the most usual initial sequence chosen by the majority of previous students -- by initially assigning weights to each column open to the student. These weights would correspond to an "optimism" factor for the a priori estimation of what the student is most likely to do. The effect of this factor could then be diminished, i.e., apply damping to the estimates as the game proceeds, thereby reducing to the standard technique.

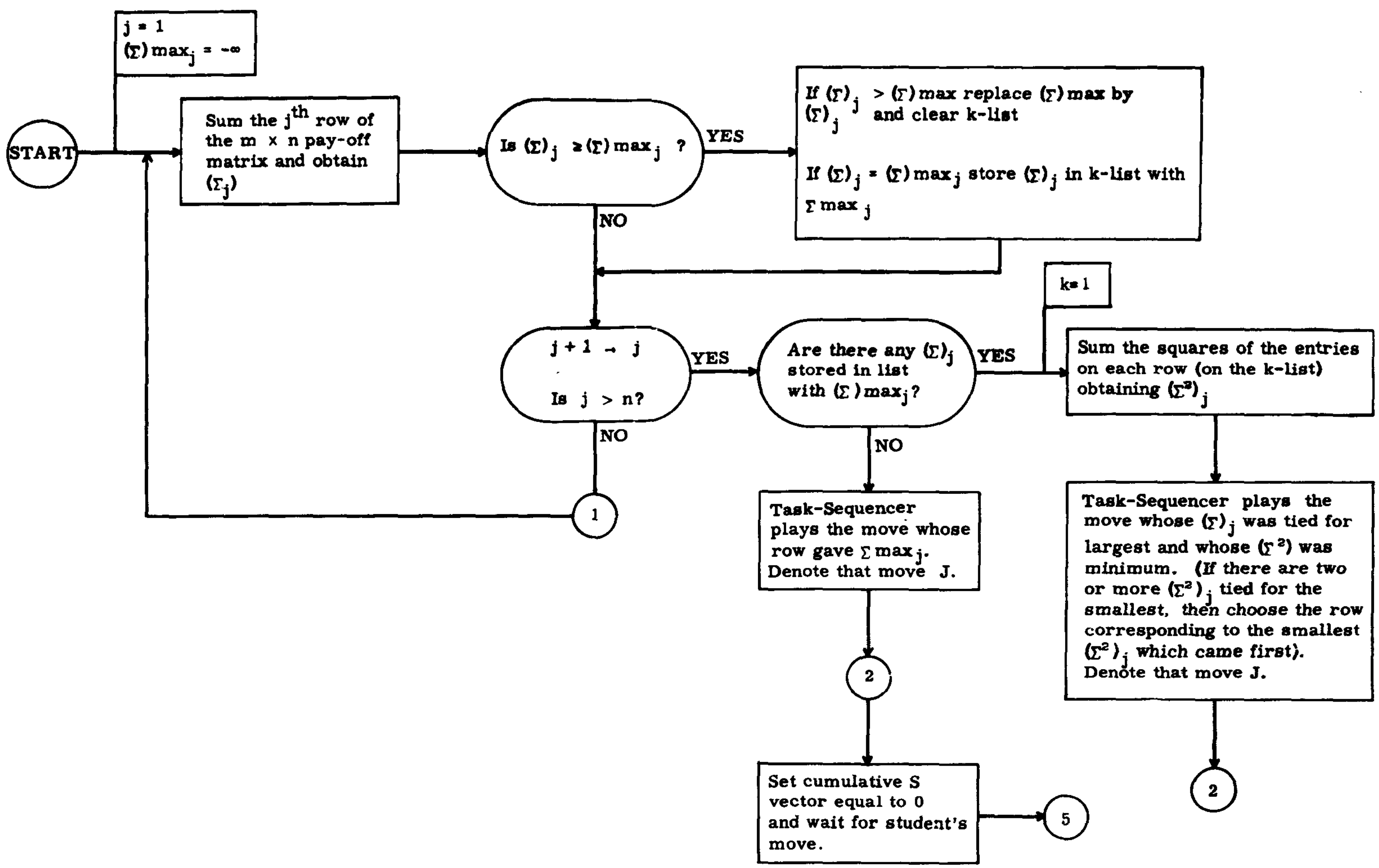
Note, that if one were to divide each entry of the cumulative vector by the number of moves, or plays, already made (equivalent to the number of additions to the vector), then the mean value of each entry would be obtained. However this averaging is unnecessary since division by a scalar does not change the relative ordering of the entries, and thus does not effect the choice of the Task-Sequencer's next move.

Instead of this simple averaging or smoothing of values, another scheme that might be optimum for the selection of the Task-Sequencer's moves is time-adaptive smoothing. That is, the influence, or weight, of the last moves or plays of the student would be greater than his first few moves. This would permit a greater sensitivity to the student's changing strategy. It would also make bluffing on the part of the student more worthwhile.

EXAMPLE FOR TACTIC-TEACHING MODE

The following specific example, although simple, illustrates the tactic-teaching mode of the Task-Sequencer. Consider a (static) situation represented by the following 4 x 3 pay-off matrix.

	K_1	K_2	K_3
J_1	3	-1	-1
J_2	-2	+1	+1
J_3	-2	4	-1
J_4	2	-3	0


START
j = 1
(Σ) max_j = -∞
Sum the j^th row of the m × n pay-off matrix and obtain (Σ_j)
Is (Σ)_j ≥ (Σ) max_j ?
YES
NO
If (Σ)_j > (Σ) max replace (Σ) max by (Σ)_j and clear k-list
If (Σ)_j = (Σ) max_j store (Σ)_j in k-list with Σ max_j
j + 1 → j
Is j > n?
YES
NO
1
Are there any (Σ)_j stored in list with (Σ) max_j?
YES
NO
k = 1
Sum the squares of the entries on each row (on the k-list) obtaining (Σ²)_j
Task-Sequencer plays the move whose row gave Σ max_j. Denote that move J.
Task-Sequencer plays the move whose (Σ)_j was tied for largest and whose (Σ²) was minimum. (If there are two or more (Σ²)_j tied for the smallest, then choose the row corresponding to the smallest (Σ²)_j which came first). Denote that move J.
2
Set cumulative S vector equal to 0 and wait for student's move.
5
2

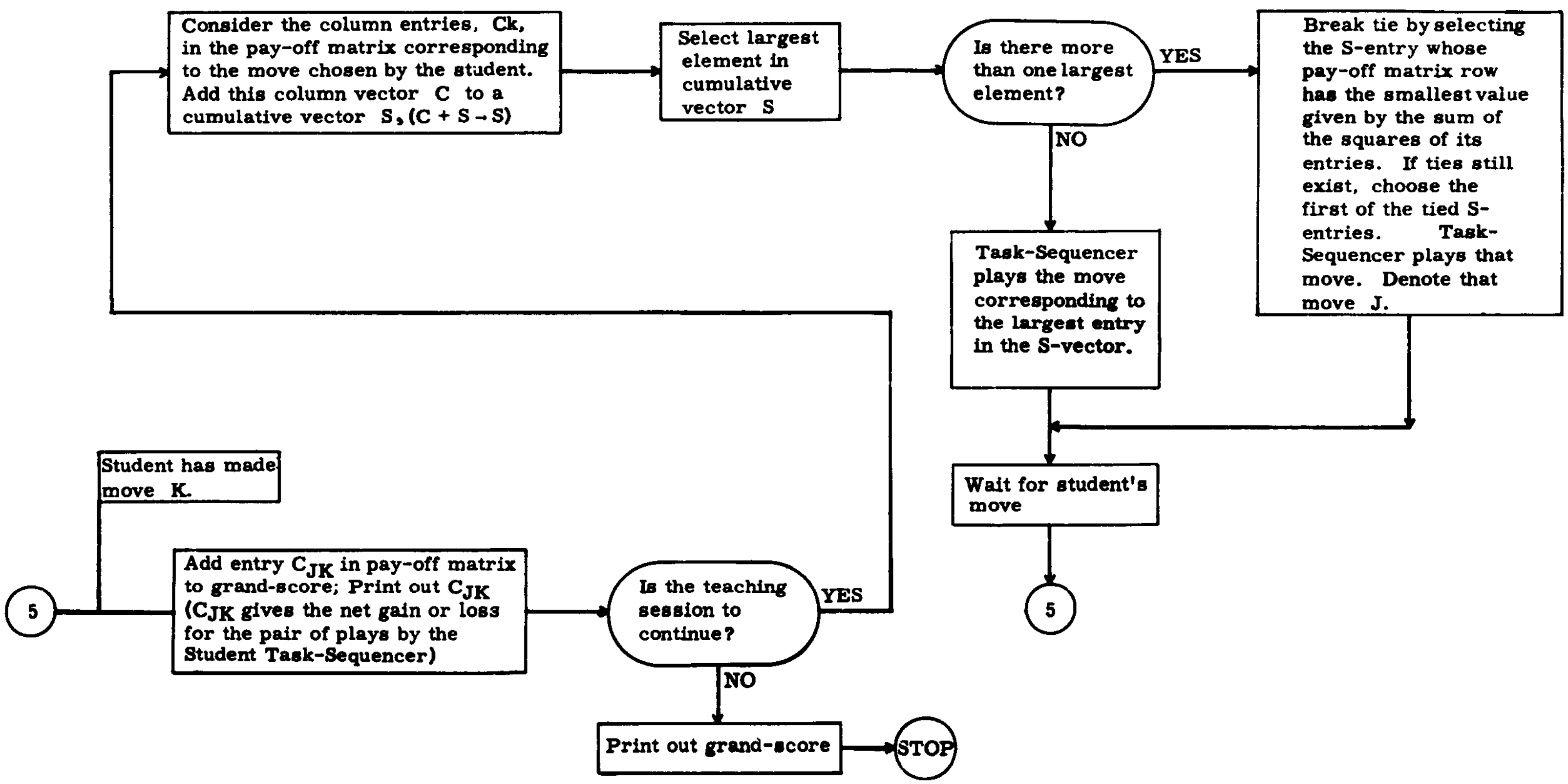


Figure 20 - Flow Chart for Tactic-Teaching Mode of Task-Sequencer

The rows J_1, J_2, J_3, and J_4 represent the four choices of strategy open to the Task-Sequencer, and the columns K_1, K_2, and K_3 represent those of the student.

To compute the first play of the Task-Sequencer, each row is summed, giving for the sum vector: (+1, 0, +1, -1).

According to the result, and the rule of initially selecting the row exhibiting the maximum expectation, J_1 and J_3 are tied.

To break the tie, the sum of the squares are now computed for J_1 and J_3 which give 11 and 21 respectively. Thus, selecting the row with the smallest variance, the first move of the Task-Sequencer is J_1. The moves are given in the form of a couplet, (J_p, K_r). The full representation for move i is

$$i: (J_p, K_r) : v_i : \Sigma v_i$$

v_i denotes the pay-off for the ith move with choice of row J_p and column K_r, and Σv_i denotes the grand score. The moves of the student were arbitrarily made by some interested person on an intuitive basis.

1: (J_1, K_3) : -1 : -1 : (-1, +1, -1, 0) cumulative vector

For the second and succeeding moves, the rule of selecting the move corresponding to the maximum entry of the (cumulative) sum of the K-vectors is followed, namely (in this case) J_2.

Thus, the game continues

2: (J_2, K_1) : -2 : -3 : (2, -1, -3, 2)

J_1 and J_4 are tied for the maximum entry in the cumulative vector. Since the sum of the squares of row 4 is 13, and only 11 for row 1, row 1 has the smallest variance and is thus chosen.

3: (J_1, K_1) : +3 : 0 : (5, -3, -5, 4)

Now the first entry is the largest one, so once again the first move is played.

4: (J_1, K_3) : -1 : -1 : (4, -2, -6, 4)

Once again a tie exists between J_1 and J_4. As before, J_1 is chosen, for it has the smaller variance of the two.

5: (J_1, K_3) : -1 : -2 : (3, -1, -7, 4)

6: (J_4, K_1) : +2 : 0 : (6, -3, -9, 6)

and so on.

DISCUSSION OF VALUE (OR PAY-OFF) MATRIX

As previously mentioned, the entries comprising the pay-off matrix are, for a realistic representation in the general case, nonstationary distributions. Therefore, it is generally necessary to modify these distributions after each task cycle -- choice of move or play by the student and Task-Sequencer. However, the required processing time to effect this by a straight-forward formula evaluation may exceed the strict real-time constraints imposed by the training situation. This is true even if the mn entries (for a m x n pay-off matrix) are independent of one another and are merely functions of time. However, the most probable situation is that the entries are not only functions of time, but are also functions of one another and their previous states.

For example, it is reasonable to suppose that there is some time decay function operating upon the worth of certain tactics, irrespective of the particular counter action taken. In penetration tactic considerations, evasive maneuvering, or feints toward other targets, is probably more dangerous to the success of the mission (i.e., costly) as the target is approached. In the example given, if some fuel has been lost or if an alternate target is assigned, the cost of maneuvering increases at a much faster rate as the target is approached. In other words, the decay function strongly depends on the previous states and perhaps the particular values or forms of the entries. In the general case, determining and calculating these entries via formula evaluation requires dealing with a number of points or distributions; a difficult, time-consuming task.

The problem of modifying the entries seems capable of being handled by employing heuristic techniques. Specific heuristics, of course, must be determined for each application. From visits to certain U.S. Air Force bases it was determined that a set of heuristics to modify the pay-off matrix representing the ECM and ECCM combat situation could be developed at the cost of extensive study. Much of the information to develop the pay-off or value matrix for that model is in the highly classified SAC tactical doctrine, but there probably would be no direct need to resort to that material. An adequate model and set of heuristics could be developed by studying some of the "war-games" and training studies already conducted by the air force.

The determination and construction of these specific heuristics are outside the scope of this study. However, a general discussion and schematic of the information flow for a heuristic technique in which these heuristics are to be incorporated are presented below. The functional description is couched in terms applicable to a wide class of models.

DISCUSSION AND SCHEMATIC FOR HEURISTIC PROGRAM

Heuristic procedures have been applied with success to varied problems, e.g., chess, checkers, theorem-proving, cryptograms, etc. Although these programs differ considerably, there is still a common structure to these techniques. Figure 21 shows the general structure of a heuristic program. This is now discussed with attention to the specific application of employing a heuristic scheme to modify the pay-off matrix.

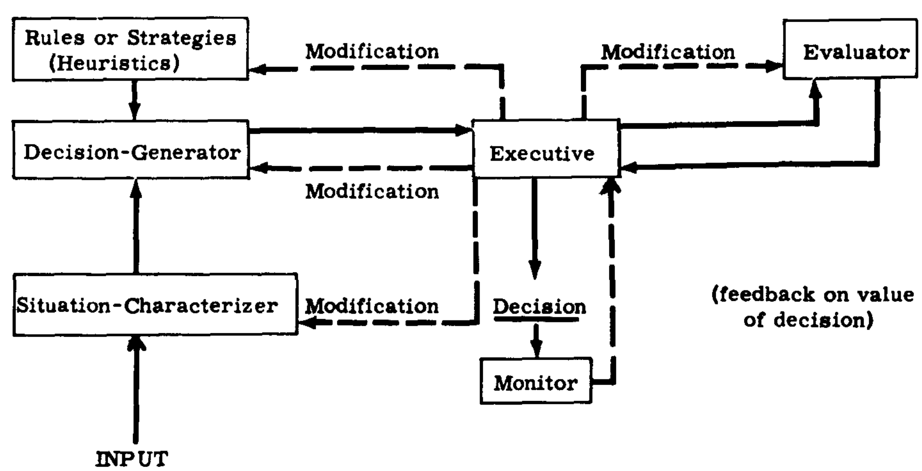


Figure 21 - General Structure of a Heuristic Program

There is some routine that characterizes the situation or position. For example, in a geometry theorem-proving computer program, a problem might be characterized by a number of questions: Does the hypothesis contain the symbol " $\|$ " ? Does the consequence of the theorem contain a statement about angles? Is the theorem concerned with arc length of circles? In a chess-playing program, the questions or characterizations might revolve about matters such as: Is it an "open" or "closed" position? Is white material ahead? Does white have two bishops against black's two knights? These qualities can be expressed in a characterization vector: 1 if yes; otherwise 0. This vector format is useful in presenting situations in terms of well-defined characteristics to the remainder of the heuristic program. It serves to abstract the (by definition) "essential" constituents of the situation to permit the program to operate upon these elements.

Essentially it creates equivalence classes for sets of inputs, and therefore the heuristic program can now consider the vector as representing the situation, or class of situations, without further recourse to the input (the situation).

For the modification of the pay-off matrix, the factors determining the situation or position, and which supply the data for the questions which develop the characterization vector, are: the matrix itself, the choice and results of the last strategy, the score of the student, and the number of plays or cycles performed. The characterization of the position, i. e., forming a characterization vector, would involve questions of these factors such as: how much fuel does the vehicle now have? (given by the number of cycles or time already spent on the "mission" and some given fuel-consumption rate); are any entries, the mean if the entry is a distribution, greater than some threshold? what is the variance of certain distributions whose mean passed a given threshold?

Another routine provides the heuristics or general strategies, which are the core of the program. These are ordered for application to the situation according to the characterization of the situation or position by a decision-generator routine. For example in the geometry theorem-proving program, if two angles must be shown to be equal and parallel lines exist, then theorems (already proven) about parallel lines and transversals might be invoked. For the chess-playing program, a position characterized by both sides having "bishops of opposite color" would cause a more aggressive speculative variation to be considered.

For the modification of entries, information, in the vector given by the situation-characterizer, that the fuel is quite low would lead to increased emphasis on situations that would have to, or could, be met by maneuvering on the part of the student. Evaluation of the terrain on which the simulated attack is taking place and the vehicle behavior would cause changes in the various tactics available. For example, greater reliance could be placed on certain radar-directed missiles if the flight vehicle is within certain elevation ranges. This fact would be reflected in the changes in the pay-off matrix.

The master executive routine would channel the set of ordered decisions (moves or statements) to a routine that serves as an evaluator. This routine, as the name suggests, evaluates the consequence of the set of decisions under question and according to some criteria selects the most promising decision. This may be done by ascribing a numerical weight to certain criteria and scoring the decision as a function of how well it satisfies them -- by means of this weighted score. Perhaps a policy of minimizing the maximum loss might guide one in setting up the criteria and the associated weights. The executive routine would select the decision with the maximum weight or, to save computing time, select the first decision whose score is above a certain threshold. In a chess-playing program one important criteria would be the material left at the end of a sequence of moves the machine would want to maximize the ratio of "value of machine's forces" to "value of opponent's forces." In a theorem-proving machine one evaluation criteria could consist of the number of theorems, which are applicable to the problem and which have already been proven, that can now be invoked.

In the tactic-teaching operation the decrease of variance in the distributions would lead to more credence on the representation of them by their mean. Decisions could be made for the remainder of the training session that certain distributions would be considered as mutually independent. Perhaps one entry is a joint distribution of three or four other entries. If various heuristics are applied to the pay-off matrix and the last response, there may be evidence that it can be considered as a function of only one entry, without loss of too much realism. However, there may be two alternate choices for selection of the independent variable (entry). Thus, which is the best choice would have to be evaluated.

This heuristic process would continue until the theorem is proven, or the chess game is won, or until the tactic-teaching session is concluded.

The various tactics or heuristics would be modified according to the results that are achieved; the several situation characterization questions or qualities could also be modified, again according to the past history; likewise, there is provision for the decision generator to have its ordering rules modified; and certainly, the evaluation-criteria and the associated weights should change as a function of experience. All of these possible modifications would be caused by reactions to the data environment and to the results achieved by present tactics. This forms a type of learning which would provide the heuristic program with greater ingenuity.

The purpose of the present study project was to ascertain the feasibility and desirability of applying heuristic programming techniques to relieve and aid the instructor in flight simulator trainers.

Presently the primary function of flight simulators is the development of student skill in procedural operations. It was determined after extensive study that there would be little advantage in employing a heuristic program , in the sense of Newell, Shaw and Simon, to sequence tasks. Instead, an algorithm was developed, that satisfies the present scheduling requirements and which is "adaptive" in the sense that it can follow and take into account the trends of the students' responses.

Furthermore this algorithm provides for the inclusion of specific heuristics if deemed desirable for some future purpose. Both single tasks and multitasks can be presented to the student with automatic scoring of the response.

There is still need of empirical experimentation with the technique, and as a result of the experimentation, there may very well be modifications to increase the flexibility of the algorithm. For example, the time spacing between task presentations has not been included in the algorithm developed. This could easily be handled by a program subroutine with fixed, random, or instructor controlled spacing.

Another application of a Task-Sequencer discussed in this report is teaching tactical operations. So far with present training methods there has been insufficient stress on this mode of operation.

An algorithm has been formulated for handling this training situation, and two possible models were discussed in a cursory manner; however, much work remains to be done. It is recommended that the ECM-ECCM tactical war game be investigated in detail to provide a realistic model for training in SAC penetration tactics. It is necessary, of course, for a detailed model to be prepared before any heuristics can be developed for dynamic modification of the representative pay-off matrix.

In addition to the conclusions and recommendations discussed above, consideration was given to several additional areas of interest. However, due to the limitations of the scope of this program, they were not studied extensively. These areas are included in the following discussion for the purpose of giving direction for future efforts.

The ultimate goal in the design of a flight trainer is to have a flight simulator system whereby it is possible to readily specify flight vehicle characteristics, environment conditions, and domain of tasks.

Of necessity, present flight trainers are extremely complicated in order to conform to an adequate level of simulation. Some of the more advanced flight simulator systems even attempt to train student crews by featuring integrated systems consisting not only of a simulated vehicle but external mechanisms such as radar.

However, as the complexity of the simulation increases, the difficulty and problems of flight instruction increase at a faster rate. Furthermore, there is insufficient precise information and data on what exactly happens in certain critical emergency situations, even with standard flight vehicles. It is difficult to predict a complete syndrome of specific aircraft malfunctions that are conditional upon one another. Even in aircraft with well established characteristics, no one is absolutely positive of the full implications of an engine-fire with respect to generated trouble symptoms. One would strongly believe that the various subsystems of the vehicle would have a whole spectrum of possible interactions depending upon the state of the subsystems and the particular emergency, in electrical engineering terms, much in the nature of a sequential circuit. Presently, the flight instructor, concurrent with his teaching sessions, must estimate the full results of each emergency and (usually) manually rotate dials to cause various malfunctions to appear on the student pilot control panel.

The problem of simulating adv[illegible]ght vehicles in ill-defined data-environments is now superimposed upon the already con[illegible] task of scheduling the sequencing of tasks to the student, and presenting to him appropriate malfunction signs of these tasks.

The present study provides for the dynamic scheduling of flight tasks, based upon the previous responses of the student (crew). This automatic scheduling of tasks leads to a more efficient training program.

To relieve further the burden on the flight instructor, it is necessary to provide him with a flight simulator system capable of operating in a probabilistic vein in contradistinction to a rigid fully predetermined and preset mode. By that is meant the flight simulator must be able to accept estimates of frequency distributions of the data environment characteristics in lieu of exact knowledge. As more experience and knowledge are gained about the environment, this information must be incorporated in the general simulation model and the distributions modified accordingly. By this means it will be possible to train students to operate vehicles in environments where only a small amount of experience is available from small samplings and to modify the simulation quickly and easily to reflect the information gained.

The first approach for such a study would be to design a simulator system with the characteristics discussed above, around one "general" class of flight vehicles, for certain suggested subsystems, which would provide for stochastic distributions equivalent to the known characteristics of "real-life" situations. Some distributions that would be necessary to have are step-functions, including the uniform distribution as a special case, and the class of exponential functions such as the normal, the Poisson, and the Rayleigh distributions.

Whether it would be best to have a functional evaluation for these distributions every cycle, or to store them internally in histogram form would depend upon the rest of the systems and would have to be determined.

It would be possible to have these distributions modified automatically (internally) even when the simulator system is given new information on the nature of the environment by qualitative remarks. For example it would be possible and desirable to allow the flight designer or instructor to communicate the fact that there is "more radiation in the Van Allen belt than previously thought," by merely stating "INCREASED RADIATION". The simulator would then modify the distribution for the radiation factor by translating the mean, increasing the variance, and biasing the distribution to the right. Naturally these qualitative remarks would have to be made in a stylized manner. If more information is known, an exact distribution can be specified by the instructors and accepted by the system.

In addition there should be various checks in the system to insure the validity of the simulation, i.e., to prevent impossible situations developing. Eventually a program generator will be needed to produce computer program models of flight simulators. With such a system (initially designed for a particular computer - say an analog-digital computer, e.g., UDOFT) it would be an easy matter to quickly change and modify characteristics of any particular flight vehicle being simulated. This would also seem to call for the development of an effective syntax for a flight simulator oriented language for instructor-simulator communications.

The simulation limits should be set as general and flexible as possible. The resulting system should be integrated with the algorithms currently developed for the dynamic scheduling of tasks to the student. This would permit not only an automatic selection of tasks but also the appropriate presentation of them.

BIBLIOGRAPHY

1. Newell, A., J. Shaw, and H. Simon, "A Variety of Intelligent Learning in a General Problem Solver," Self-Organizing Systems, Pergamon Press, N.Y., 1960, pp. 153-189.

2. Brown, G. W., "Iterative Solutions of Games by Fictitious Play," Activity Analysis of Production and Allocation, (Cowles Commission for Research in Economics, Monograph No. 13, eds. Koopmans, Tjalling C. et al.), John Wiley & Sons, Inc., N.Y.C., 1951.

3. Robinson, J., "An Iterative Method of Solving a Game," Annals of Mathematics, No. 54, 1951, pp. 296-301.

4. Miller, R. B., Handbook on Training and Training Equipment Design, WADC Technical Report 53-136, Wright Air Development Center, Wright-Patterson Air Force Base, Ohio, 1953.

5. Air Force Manual 51-40, Air Navigation, Volume I, Department of the Air Force, U. S. Government Printing Office, Washington, D.C., 15 October 1959.

6. USAF Series KC-135 Aircraft Flight Manual, Technical Order 1C-135(K) A-1, Official USAF Publication, March 1, 1960.

7. All Weather Flight Manual, Manual Navaer 00-80 T-60, Issued by Office, Chief of Naval Operations, Reproduced by FAA. Bur. of Flight Standards, 1957.

8. B-29 Aircraft Mechanics and Maintenance Course, Training Manual, Superfortress School, 3094th AAF Base Unit, Seattle, Washington, June 1945.

9. Muckler, F. A., Psychological Variables in the Design of Flight Simulators for Training, WADC Technical Report 56-369, Wright Air Development Center, Wright-Patterson Air Force Base, Ohio, 1959.

10. Systems Operational Performance Requirements, F-105 Flight Simulator, Nuclear Products Report No. 44-8004-105, June 1959.

11. Technical Manual, KC135A Trainer, Flight Simulator, Technical Order 43D3-3-8-11, Official USAF Publication, April 26, 1960.

12. Technical Manual, Simulator Trainer, B52 Aircraft, Type AF/F37A-T1, Technical Order 43D3-2-5-11, Official USAF Publication, January 1, 1960.

13. Universal Digital Operational Flight Trainers, Report No. 57-02, Moore School of Electrical Engineering, University of Pennsylvania, Phila., Pa., 1956, RSTLA No. AD115792.

APPENDIX

LIST OF ABBREVIATIONS

C/B	-	Circuit Breaker
Sw.	-	Switch
D/L	-	Display Light
EOP	-	End of Problem
R/C	-	Rate of Climb
D/I	-	Direction Indicator
C/O	-	Cut off
EGT	-	Exhaust Gas Temperature
C/M	-	Crew Member(s)
TAS	-	Turn and Slip Indicator
EPR	-	Engine Pressure Ratio
P	-	Pilot
CP	-	Copilot
A/S	-	Air Speed Indicator
I	-	Instructor
C/L	-	Check List
F/D	-	Fuel Dump
CP/R	-	Copilot Reads Check List
A/R	-	Air Ram
S/E	-	Smoke and Fumes Elimination
E/D	-	Emergency Descent
B/O	-	Bail Out
EA/S	-	Engine Air Start

Aerospace Medical Division,
6570th Aerospace Medical Research Laboratories, Wright-Patterson AFB, Ohio
Rpt. No. AMRL-TDR-63-17. DYNAMIC TASK SCHEDULING IN FLIGHT SIMULATORS. Final report, Feb 63, iv + 45 pp, incl. illus., table, 13 refs. Unclassified report

This report deals with the possible mechanization of dynamic task scheduling in flight simulators, i. e., developing a Task-Sequencer. Attention is focused on the possible application of some of the heuristic programming techniques and an evaluation of their worth for that specific purpose is made. Two main applications for a Task-Sequencer are defined. The first involves the traditional training of students (flight crews) for flight vehicle operation, termed (over)

the operation-teaching mode. The second is for the development of tactical skill, i. e., crew decision-making capabilities, termed the tactic-teaching mode. Algorithms for task sequencing in real time are formulated for both of these classes of applications. The only possible benefits in employing a heuristic programming scheme appear to exist when it is used for an ancillary role in the tactic teaching mode. This includes development of specific task flow diagrams and associated scoring charts. Finally, recommendations are made for further work.

UNCLASSIFIED

1. Flight Simulators
2. Training and Training Aids
3. Procedural Tasks
4. Simulation (Aerodynamics)

I. AFSC Project 6114, Task 611409
II. Behavioral Sciences Laboratory
III. Contract AF 33(616)-8062
IV. Burroughs Corporation, Burroughs Laboratories, Paoli, Pennsylvania

UNCLASSIFIED

UNCLASSIFIED

V. J.M. Kurtzberg
VI. In ASTIA collection
VII. Aval fr OTS: $1.50

UNCLASSIFIED

Aerospace Medical Division,
6570th Aerospace Medical Research Laboratories, Wright-Patterson AFB, Ohio
Rpt. No. AMRL-TDR-63-17. DYNAMIC TASK SCHEDULING IN FLIGHT SIMULATORS. Final report, Feb 63, iv + 45 pp, incl. illus., table, 13 refs. Unclassified report

This report deals with the possible mechanization of dynamic task scheduling in flight simulators, i. e., developing a Task-Sequencer. Attention is focused on the possible application of some of the heuristic programming techniques and an evaluation of their worth for that specific purpose is made. Two main applications for a Task-Sequencer are defined. The first involves the traditional training of students (flight crews) for flight vehicle operation, termed (over)

the operation-teaching mode. The second is for the development of tactical skill, i. e., crew decision-making capabilities, termed the tactic-teaching mode. Algorithms for task sequencing in real time are formulated for both of these classes of applications. The only possible benefits in employing a heuristic programming scheme appear to exist when it is used for an ancillary role in the tactic teaching mode. This includes development of specific task flow diagrams and associated scoring charts. Finally, recommendations are made for further work.

UNCLASSIFIED

1. Flight Simulators
2. Training and Training Aids
3. Procedural Tasks
4. Simulation (Aerodynamics)

I. AFSC Project 6114, Task 611409
II. Behavioral Sciences Laboratory
III. Contract AF 33(616)-8062
IV. Burroughs Corporation, Burroughs Laboratories, Paoli, Pennsylvania

UNCLASSIFIED

UNCLASSIFIED

V. J.M. Kurtzberg
VI. In ASTIA collection
VII. Aval fr OTS: $1.50

UNCLASSIFIED